Quality of Service i IPv6 og ATM

Hovedoppgave
ved
sivilingeniørutdanning i
informasjons- og kommunikasjonsteknologi

av
Vidar Karlsen

Grimstad, juni 1999

Forord

Denne rapporten inneholder resultater fra diplomarbeidet utført vårsemesteret 1999 av Vidar Karlsen, som er student ved sivilingeniørstudiet i informasjon og kommunikasjonsteknologi ved Høgskolen i Agder.

Oppgaven har blitt gitt av Statoil IT i Stavanger, som tar sikte på å integrere tjenester som data, tale, video og telemetri i ett nett. Det var fra Statoil side et ønske om bl.a. å utrede hvordan teknologiene ATM og IPv6 kan garantere tjenestekvalitet.

Med bakgrunn i min interesse i datakommunikasjon, var dette en oppgave jeg syntes var veldig interessant. Dette området er i sterk utvikling og jeg ville med denne oppgaven, være attraktiv på arbeidsmarkedet, ettersom det er få i Norge som har inngående kunnskap på dette området.

Jeg vil først og fremst takke veileder Nils Ulltveit-Moe (Høgskolen i Agder) for god veiledning og oppfølging, samt Ole Petter Drange (Statoil IT – Stavanger) for oppgaven og mange gode ideer og råd underveis. Jeg vil også takke Peter Gustafsson og Vidar Bakken (Cisco Systems – Oslo) for besøket ved Cisco, samt god hjelp til testingen.

Grimstad, 28 Mai 1999

Vidar Karlsen

Sammendrag

Hovedoppgaven som er beskrevet i dette dokumentet, besvarer en oppgave som er gitt av Statoil i Stavanger. Statoil har i løpet av de ti siste årene bygget opp forskjellige typer nett for overføring av data, tale, video og telemetri. Det virker nå som om den teknologiske utviklingen innen Internet-teknologier kan muliggjøre en integrasjon av ovennevnte tjenester ved å benytte IP protokollen.

Med utgangspunkt i Statoil og IT som visjon, er det viktig at dette nettverket til en hver tid gjenspeiler de forventninger Statoil har. Det er derfor viktig å få svar på hvordan Quality of Service kan garanteres i IP.

Dette dokumentet beskriver hvordan teknologiene, IPv6 og ATM, er med på å garantere QoS. IPv6 er en ny versjon av den gamle Internet Protokollen (IP), der vi finner forbedringer som *utvidede adressemuligheter* og *merking av flyt og prioriteringer*.

ATM (Asynchronous Transfer Mode) er en forbindelsesorientert pakkesvitsjet teknologi som har sitt utspring i telekommunikasjonsverdenen. ATM har blitt en suksess som en link lags teknologi, fordi den tilbyr høyhastighets forbindelser til rutere og nettverk, gjennom et fleksibelt, høyhastighets og skalerbart link lag.

En enkel definisjon på QoS, er at QoS er et mål på hvor godt nettverket oppfører seg og et forsøk på definere karakteristikker og egenskaper til spesielle tjenester. I dette inngår også muligheten for å differensiere mellom trafikk- eller tjenestetyper, slik at brukere kan behandle en eller flere klasser av trafikk forskjellig.

I IPv6 er det først og fremst Traffic Class og Flow Label feltene i IPv6 headeren, som er med på å differensiere mellom trafikk- og tjenestetyper. Flow Label kan sammen med RSVP, brukes til å reservere en viss båndbredde til f.eks. en trafikk- eller tjenestetype.

En applikasjon som bruker et ATM nettverk, overbringer dens båndbredde og ytelses behov gjennom trafikk kontrakten. Trafikk kontrakten inneholder bl.a. tjenestekategori og tjenestekvalitet til den ønskede forbindelsen. I ATM har vi også mekanismer som *Connection Admission Controll* og *Trafikk tilpassing*, som er med på å reservere båndbredde til en viss tjenestekvalitet og tilpasse cellene til en sett med trafikk deskriptorer.

Det har også i denne hovedoppgaven, blitt utført måling av QoS parametere i forskjellige nettverkstopologier, samt utprøving av RSVP.

Selv om IPv6 feltene Traffic Class og Flow Label i IPv6 headeren gir rom for QoS, er det per i dag ingen kommersielle applikasjoner som støtter disse feltene. Dessuten har mye av den nye funksjonaliteten blitt implementert i den gamle versjonen.

Selv om ATM gir et stort sett med QoS knagger, er ikke teknologien allment brukt som en ende-til-ende transport protokoll. På bakgrunn av dette, er ikke ATM ideell til bruk i innføring av QoS.

Innhold

OVERSIKT OVER TABELLER

OVERSIKT OVER LIKNINGER

OVERSIKT OVER FIGURER

1 Innledning

I det siste semesteret av sivilingeniør utdanningen i Grimstad skal studentene utføre en oppgave, som kalles diplom oppgaven. Denne oppgaven, med fagkode IT6401 [1], er på 10 vekttall og blir gitt av skolen, næringsliv eller utformet personlig fra studentens faglige interesser. Oppgaven kan utføres individuelt eller som gruppeoppgave.

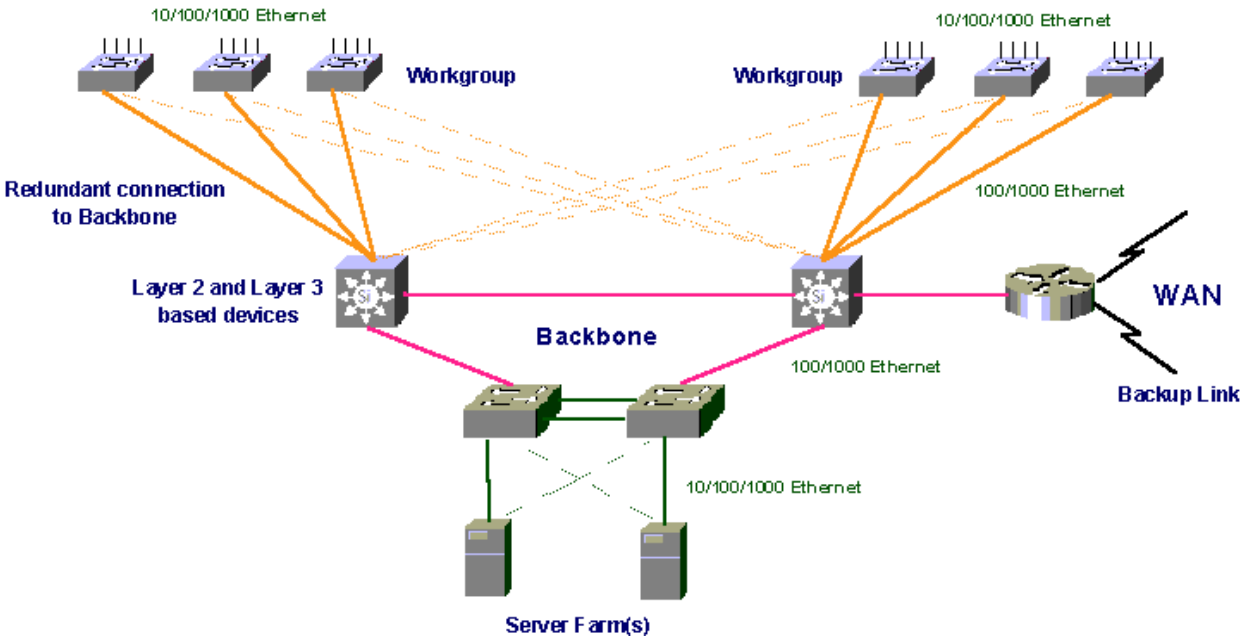
1.1 Definisjon av oppgaven

Oppgaven går primært ut på å se på hvordan IP versjon 6 (IPv6) og ATM kan garantere tjenestekvalitet (QoS – Quality of Service). IPv6 bygger på Internet Protokollen (IP) og er en videreutvikling av den. ATM (Asynchronous Transfer Mode) er en teknologi som bygger på svitsjing av pakker, som IP, og har hatt en enorm utvikling de senere årene.

I oppgaven skal det legges vekt på å beskrive konkrete og presise veiledninger for hvordan IPv6 og ATM kan garantere QoS. Med dette menes hvilke mekanismer, interne og eksterne, som er med på i gi QoS. Det er også ønskelig fra Statoils side å se på hvordan sikkerheten blir ivaretatt i disse teknologiene.

For å få bedre innsikt i hvordan mekanismene som er med på å garantere QoS fungerer, skal det utføres en test. I denne testen inngår også måling av QoS parameterene, pakketap og throughput.

Det vil også bli gjort en markedsundersøkelse på utstyr, for å finne ut hvilke leverandører/produkter som støtter QoS tjenester.



Figur 1: Oversikt over Statoils nett.

1.2 Motiv

Bruken av informasjonsteknologi er i Statoil anerkjent som et nødvendig og virksomt hjelpemiddel til å nå konsernets forretningsmessige mål og til å videreutvikle konsernet mot de fremtidige mål som Statoils visjon legger opp til. Dette gjenspeiles også i Statoil IT sin nye visjon: ”Statoil skal være verdensberømt for sine resultater på å integrere forretning og informasjonsteknologi.” Gode og effektive telekommunikasjoner gir et helt avgjørende bidrag til å kunne realisere bruken av IT på tvers av land og lokasjoner.

Statoil har i løpet av de ti siste årene bygget opp forskjellige typer nett for overføring av data, tale, video og telemetri (se figur 1). Det virker nå som om den teknologiske utviklingen innen Internet-teknologier kan muliggjøre en integrasjon av ovennevnte tjenester ved å benytte IP protokollen.

Med utgangspunkt i Statoil og IT som visjon, er det viktig at dette nettverket til en hver tid gjenspeiler de forventninger Statoil har. Det er derfor viktig å få svar på hvordan Quality of Service kan garanteres og hvordan sikkerheten kan ivaretaes i IP.

1.3 Avgrensing av oppgaven

Denne oppgaven er et resultat av en idé dugnad, som ble hold hos Statoil i Stavanger høsten 1998. Resultatet av idé dugnaden er gitt i kapittel 1.1.

Oppgaven ble følgelig ganske omfattende og jeg har underveis vært nødt til å begrense oppgaven. Jeg har i samarbeid med veileder, kommet frem til at følgende punkter skal fjernes.

- *Hvordan sikkerheten kan ivaretas i IPv6 og ATM*. Dette området er stort og kunne i seg selv, vært grunnlag for en egen hovedoppgave. Jeg valgt å se bort fra denne delen, for i størst mulig grad konsentrere meg om QoS i de nevnte teknologier. Jeg vil imidlertid vise til [2], der temaet er behandlet.

- *Markedsundersøkelse på utstyr*. Ettersom dette området er i stor utviklig, samtidig som Statoils sammensmelting av nett ikke vil skje med det første, ville en undersøkelse av produkter som finnes p.d.d., være til liten nytte. På bakgrunn av dette, samt at oppgaven fortsatt var for omfattende, valgte jeg også å se bort fra dette punktet.

- *Test av IPv6 hos Cisco Systems*. Det ble valgt å ikke satse å teste på IPv6 (jf. argumentasjon i kapittel 7.1). Grunnet mangel på applikasjoner og utstyr, ble det heller ikke målt pakketap i de forskjellige testscenariene.

1.4 Mål

Utformingen av konkrete mål for oppgaven har vært en dynamisk prosess. Med dette menes at mål og målformuleringer har forandret seg etterhvert som jeg har tilegnet meg ny kunnskap.

Mål for oppgaven er:

- Studere og få inngående kunnskap om teknologiene. Det vil si at jeg må sette meg inn i IPv4, IPv6, ATM og RSVP.
- Skal se på hvilke mekanismer som er med på å garantere QoS i IPv6 og ATM.
- Beskrive konkrete og presise veiledninger for hvordan IPv6 og ATM kan garantere Quality of Service.
- Utføre en så realistisk test som mulig på de mekanismene som blir omtalt i oppgaven.

1.5 Metode

Denne hovedoppgaven har i første rekke vært et litteraturstudie. Hele perioden, fra tidlig januar til medio mai, har gått med til å sette seg inn i teknologiene. På forhånd hadde jeg noe kunnskap om teknologiene fra fagene IT2200 (Datakommunikasjon) ved Høgskolen i Agder. Jeg hadde også noe kunnskap om hva QoS innebærer fra fagene IT2200 og IT2300 (Telekommunikasjonssystemer).

Jeg begynte med å se på grunnleggende IP [3], for så å gå videre med neste generasjons IP [3,4,5,8]. Deretter tok jeg for meg grunnleggende ATM [3].

Før jeg begynte analysen av hvordan teknologiene kan garantere QoS, satte jeg meg inn i publiserte definisjoner av QoS, for å danne meg en oppfatning av hva begrepet omfatter [6,7].

Etter analysen av QoS, begynte jeg forberedelsene til test delen. Her satte jeg meg inn i hvordan man oppgraderer klienter til IPv6, samt diverse software som skulle benyttes i testen [9,10,11,25].

1.5.1 Videre arbeid

Mye av litteraturen er modningsstoff. Etterhvert som forståelsen av teknologiene begynte å vokse, prøvde jeg å samle trådene fra den litteraturen jeg har lest, for å danne mine egne oppfatninger og konklusjoner av problemet.

Dette gjenspeiles også i rapporten, der jeg først gir en innføring i grunnleggende IPv6 og ATM, før jeg går i gang med definisjon av begrepet QoS. Etter dette kommer jeg inn på mekanismer i teknologiene som er med på å garantere QoS, før jeg til slutt kommer med en anbefaling og konklusjon.

1.5.2 Møter

Det har blitt holdt jevnlig møter mellom Nils Ulltveit-Moe, Ole Petter Drange og meg gjennom hele prosjekt perioden. Møtene har vært forsøkt holdt annenhver mandag, samt tilleggsmøter etter behov. På disse møtene har det blitt diskutert områder som jeg har undersøkt, videre arbeid og forslag til forbedringer. Møtene har foregått via videokonferanse eller telefon.

Ettersom oppgaven utføres for Statoil, som holder til i Stavanger, har det oppstått et behov for å legge dokumenter, linker og møtereferater et sted hvor alle parter har tilgang. Jeg har derfor opprettet et eget område i et *knowledge management system*, kalt WebWare. I dette verktøyet får alle parter tilgang til dokumentene som blir lagt ut, ved hjelp av Internet og WWW aksess.

2 Grunnleggende IP versjon 6

Den nye IP er basert på en enkel filosofi; at Internet ikke kunne vært så vellykket i de senere årene, hvis IPv4 inneholdt store feil. IPv4 har en veldig god design, og IPv6 skulle beholde flesteparten av dens karakteristikker.

2.1 Forandringene fra IP versjon 4

IPv6 (Internet Protocol version 6) eller IPng (IP next generation) er en ny versjon av Internett Protokollen, designet som en arvtager til IPv4. Forandringene fra IPv4 til IPv6 faller primært under disse kategoriene [5,20] :

- *Utvidede adressemuligheter*. IPv6 øker adressefeltet i IP fra 32 bits til 128 bits for å støtte flere nivåer av adresseringshierarkier, et større antall adresserbare noder og enklere autokonfigurasjon av adresser. Skalerbarheten av multicast ruting er forbedret ved å legge til et ”formål” felt til multicast adresser. Det er også blitt definert en ny type adresse, kalt ”anycast adress”, som blir brukt til å sende en pakke til hvilken som helst av en gruppe av noder.

- *Forenkling av header formatet*. Noen av feltene i IPv4’s header har blitt fjernet eller gjort valgfrie. Dette for å reduserer kosten av pakkehåndtering og for å optimalisere båndbredde bruken til IPv6 headeren.

- *Forbedret støtte for utvidelser og valg*. Forandringer i måten valgene i IP headeren er kodet, tillater en mer effektiv forwarding, mindre bindene grenser på lengden av valgene og større fleksibilitet for introduksjon av nye valg i fremtiden.

- *Mulighet for merking av trafikkflyt og prioritering*. I IPv6 har det blitt åpnet for muligheten for merking av pakker som hører til en bestemt trafikk flyt som senderen krever skal bli behandlet på en bestemt måte. For eksempel en gitt tjenestekvalitet som går utenom standard eller real-time tjenester.

- *Gjenkjennings og personverns muligheter*. Utvidelser for å støtte gjenkjenning, integritet og konfidensialitet av data er spesifisert i IPv6.

2.2 IP versjon 6 sitt header format

IPv6 headeren består av 64 biter, etterfulgt av to 128 biters IPv6 adresser for kilde og destinasjon. Den totale lengden kommer da opp i 40 bytes. Figur 2 viser header formatet til IP versjon 6.

De første 64 bitene består av [4,5,20]:

- Version (4 biter), som forteller hvilken versjon av IP som benyttes. For IPv6 er version = 6.
- Traffic Class (8 biter)
- Flow Label (20 biter)

Version	Traffic Class	Flow Label	
Payload Length		Next Header	Hop Limit
Source Address			
Destiantion Adress			

Figur 2 : IP versjon 6 headeren

- Payload Length (16 biter), som angir lengden på payloaden, d.v.s. resten av pakken etter headeren i oktetter.
- Next Header (8 biter), som identifiserer headertypen som følger straks etter den etterfølgende IPv6 headeren.
- Hop Limit (8 biter). Reduserer verdien med 1 for hver node som forwarder pakken. Pakken forkastes hvis Hop Limit når verdien 0.

2.2.1 Sammenlikning av to headere

IPv6 headeren er faktisk ganske mye enklere en sin forgjenger, IPv4. Hvis vi ser på antall felt som er med i IPv6 headeren kan vi bare telle seks felt og to adresser. Mens den gamle headeren har ti faste felt, to adresser og noen valg. IPv4 headeren er vist i figur 3.

Version	IHL	Type of Service	Total Length	
Identification			Flags	Fragment Offset
Time to Live		Protocol	Header Checksum	
Source Address				
Destination Address				
Options				Padding

Figur 3: IP versjon 4 headeren

Det eneste feltet som har beholdt den samme meningen og den samme posisjonen er versjon nummeret, som i begge tilfeller blir kodet i de første fire bitene i headeren. I utgangspunktet var det meningen å benytte IPv4 og IPv6 samtidig på de samme kablene, på de samme lokale nettverkene, bruke de samme innkapslingene og de samme link driverne. Nettverks programmet skulle bruke de første fire bitene, som er versjonsfeltet, til å bestemme pakkens videre prosessering. Hvis versjonsfeltet inneholdt 4 (0100 på binær form), ble pakken gjenkjent som en IPv4 pakke og hvis versjonsfeltet inneholdt 6 (0110 på binær form), ble pakken gjenkjent som en IPv6 pakke [5]. Denne ideen ble forkastet, eller i hvertfall lagt mindre vekt på. Når det er mulig, skal IPv4 og IPv6 bli demultiplekset ved medie laget. For eksempel vil IPv6 pakker bli bert over Ethernet med innholds typen 86DD (hexadesimalt) i steden for IPv4's 8000.

Følgende seks felt ble utelatt:

- Header Length
- Type of Service
- Identification
- Flags
- Fragment Offset
- Header Checksum

Følgende tre felt ble omdøpt og i noen tilfeller en smule omdefinert:

- Total Length
- Protocol
- Time to Live

Opsjons mekanismen ble helt revidert og to nye felt ble lagt til:

- Traffic Class
- Flow Label

2.2.2 Forenklinger

IPv4 headeren ble basert på ”state of the art” fra 1975. En bør derfor ikke være overrasket over, at en 20 år etter, kan gjøre følgende tre store forenklinger:

- Tildele et fast format for alle headere.
- Fjerne ”Header Checksum”.
- Fjerne hop-by-hop segmenterings prosedyren.

IPv6 headeren inneholder ikke noen valgfrie elementer. Dette betyr ikke at vi kan uttrykke opsjoner for særegne pakker. Som en vil se i neste kapittel, blir ikke dette utført i et opsjonsfelt av variabel lengde som i IPv4. I steden blir det føyet til tilleggsheadere etter hoved headeren [5]. En får da følgelig ikke bruk for et felt, som indikerer lengden på headeren (IHL) i IPv6.

Å fjerne ”Header Checksum” kan virke som et dristig trekk. Hovedfordelen er å forminske kosten ved header prosessering, fordi det ikke er nødvendig å sjekke og oppdatere checksum-men ved hvert skift. Den åpenbare risikoen blir at uoppdagede feil fører til at pakker kommer feil [4]. Denne risikoen er imidlertid minimal, ettersom de fleste innkapslings rutiner inneholder en checksum.

IPv4 inneholdt en fragmenterings prosedyre, slik at senderen kunne sende store pakker uten å bekymre seg over kapasiteten til skiftene. De store pakkene vil bli delt opp i passende frag-menter, hvis det trengtes. Mottakeren vil vente på alle disse fragmentene og rekonstruere pakken. En lærte imidlertid en viktig lekse fra erfaringen med transport kontroll protokoller, nemlig at enheten som sender bør også ha kontrollen [3].

Anta at vi prøver å overføre store pakker over et nettverk som bare kan ta små segmenter. For at overføringen skal være vellykket, må alle segmentene komme over uten feil. Hvis bare et segment mangler må en overføre hele pakken på nytt, noe som resulterer i en ueffektiv bruk av nettverket.

Regelen i IPv6 er at senderen skal lære seg den maksimale segmentstørrelsen gjennom en prosedyre som kalles ”Path MTU Discovery” [4,5]. Hvis en prøver å sende større pakker, vil disse pakkene bli avvist av nettverket. Som en konsekvens av dette er det ikke noe behov for noen segmenterings kontroll felt i IPv6, som det er i IPv4. Disse feltene er Identification, Flags og Fragment Offset.

IPv6 inneholder imidlertid en ende-til-ende segmenteringsprosedyre. Alle IPv6 nettverk skal angivelig være i stand til å bære en payload på 536 oktetter. Noder som ikke vil finne eller huske linkens MTU kan bare sende små pakker.

Den siste forenklingen i IPv6 er fjerningen av Type of Service feltet. I IPv4 ble TOS feltet brukt til å indikere preferansene for korteste, billigste (lavest kost) og den sikreste banen. Imidlertid ble ikke dette feltet brukt så ofte av applikasjonene.

2.2.3 Reviderte IPv4 parametere

Som i IPv4, inneholder IPv6 headeren Total Length, Time to Live og Protocol. Imidlertid ble definisjonen av disse feltene revidert i lys av erfaring.

Total Length feltet i IPv4 er byttet ut med Payload Length i IPv6. Det er en hårfin forskjell på disse to formuleringene, fordi med Payload Length mener en lengden på dataene etter headeren.

Anta for eksempel at payloaden er en TCP pakke bestående av 20 byte TCP header og 400 byte data. I IPv4 ville vi lagt til en 20 bytes IPv4 header foran denne TCP pakken slik at den totale lengden ble 440 byte. I IPv6 ville vi lagt til en 40 byte IPv6 header, men lengden på payloaden ville bli satt til 420 byte, ikke 460. I IPv6 som i IPv4 blir lengde feltet kodet med 16 biter, noe som begrenser pakken til 64kB.
IPv6 har en forskrift for større pakker som kalles ”jumbogram” opsjonen.

Protocol feltet ble omdøpt til Next Header for å reflektere den nye organiseringen av IP pakker. I IPv4 blir headeren etterfulgt av data fra en transport protokoll, f.eks. TCP eller UDP pakker. IPv6 har akkurat den samme strukturen, noe som gjør at Next Header feltet indikerer protokollene TCP eller UDP. I IPv6 kan en også skyte inn tilleggsheadere mellom IP og TCP eller UDP payloaden. Next Header feltet vil da bli satt til typen til den første tilleggsheaderen.

Time to Live (TTL) feltet er omdøpt til Hop Limit. I IPv4 ble TTL feltet utrykt i antall sekunder, for å indikere hvor lenge pakken kan leve i nettverket før den blir tilintetgjort. Forestillingen rundt TTL var basert på teoretisk analyse av transport kontroll protokoller [3]. Hvis pakker fikk lov til å oppholde seg i nettverket i det uendelige, kunne gamle kopier dukke opp og skape feil i protokollen [4]. TCP har for eksempel et begrep om at en forbindelse skal gå på ”tomgang” i en kort periode etter at forbindelsen er avsluttet, for å sikre at pakker som hører til den gamle forbindelsen blir slettet fra nettverket. Denne mekanismen fungerer bare når transport protokoll mekanismene vet hvor lenge en pakke kan oppholde seg i nettverket, noe som er formålet med TTL feltet [3].

IPv4 spesifikasjonen krever at TTL feltet blir dekrementert med 1 sekund eller tiden den oppholder seg i ruter køen, hvis tiden er større enn 1 sekund. Det er imidlertid meget vanskelig å estimere den nøyaktige tiden en pakke venter i ruteren. Siden denne tiden vanligvis blir målt i millisekunder og ikke i sekunder, dekrementerer de fleste ruterene TTL feltet med 1. Denne

oppførselen har blitt offisiell i IPv6, derav navnet Hop Limit. Dette feltet teller antall hopp, ikke antall sekunder. Transport protokoller skal skaffe egne metoder for beskyttelse mot oppkom av gamle pakker, f.eks. bruk av tidsstempel.

2.2.4 Nye felter

Det er to helt nye felter i IPv6 headeren som ikke var med i IPv4, nemlig Flow Label og Traffic Class. Begge feltene er designet for å kunne håndtere real-time trafikk. Traffic Class feltet har 8 biter for at sendende noder og rutere skal kunne identifisere og skjelne mellom forskjellige trafikk klasser eller prioriteter til IPv6 pakker.

Flow Label er et 20 biters felt og blir brukt til merking av sekvenser av pakker som krever spesiell behandling av IPv6 ruterene, som f.eks. tjenestekvalitet som går utenfor normalen eller real-time tjenester.

Jeg vil komme tilbake til disse feltene under min analyse av hvordan IPv6 kan garantere tjenestekvalitet i kapittel 5.

2.3 IP versjon 6 og tilleggsheadere

IPv4 headeren hadde rom for opsjoner, noe som gjorde at noen pakker kunne få spesial-behandling. Den originale spesifikasjonen inneholder koder for sikkerhets opsjoner, kilde ruting og tids merking. Opsjoner, falt imidlertid gradvis ut av bruk, for det meste på grunn av effekten til ytelsen.

Koden for forwarding av pakker er til de grader optimalisert i software til ruterene [4]. Programmererne teller bokstavelig talt instruksjonene som trengs for å prosessere en pakke, fordi en reduksjon i antallet instruksjoner resulterer i en høyere ytelse. En ruter som kan forwarde flere pakker pr. sekund enn en konkurrerende ruter fra en annen leverandør, vil være den mest attraktive på markedet.

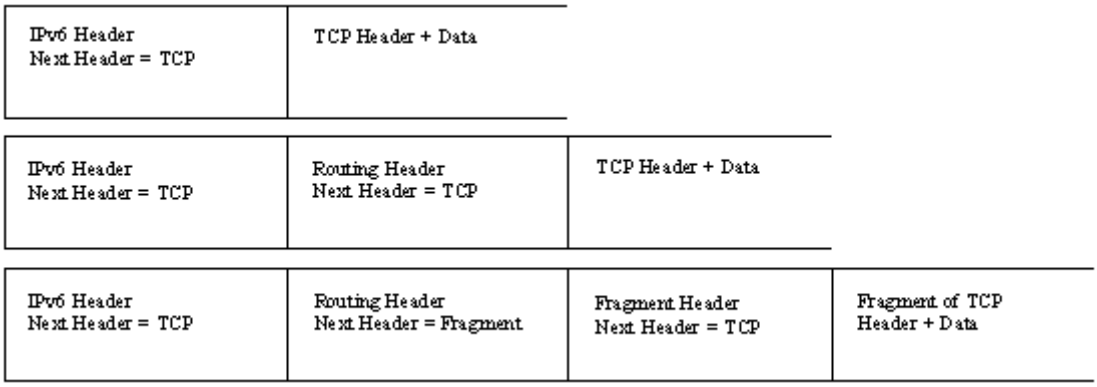
Den mest vanlige måten å øke hastigheten på, er å konsentrere på de mest vanlig pakkene, slik at de kan følge en fast rutine i programmet [4,5]. Pakker med opsjoner, kan ikke følge den faste rutinen i software fordi de trenger spesialbehandling. Ofte vil disse pakkene ganske enkelt bli plassert i en underordnet kø og bli behandlet som en ”annen rangs borger” av en tregere og mindre optimalisert software. Som et resultat av dette blir pakker som benytter seg av opsjoner straffet.

Dette har resultert i at pakker med opsjoner har blitt mindre vanlig, noe som har gjort at de som designer ruterene har mindre grunn til å lage en effektiv håndtering av pakker med opsjoner. Hvis vi følger denne tankegangen noen år, kan en like gjerne fjerne opsjonene fra IPv4 headeren, fordi ingen vil benytte dem.

Det er imidlertid god grunn til å kreve spesialbehandling av noen pakker, for eksempel hvis en vil bruke en spesiell rute, gjennom kilde ruting, eller hvis en vil spesifisere en bestemt hånd-tering av pakken til mottakeren. Etter å ha lært fra erfaringer med IPv4, spesifiserer IPv6 disse spesialbehandlingene gjennom tilleggsheadere.

2.3.1 En rekke med headere

I IPv4 følger payloaden, f.eks. en TCP pakke, rett etter IP headeren. I IPv6 er det mulig å sette inn et vilkårlig antall tilleggsheadere mellom IP headeren og payloaden. Hver header blir identifisert ved hjelp av en header type og inneholder headertypen til den etterfølgende headeren i rekken, eller til payloaden hvis det er den siste tilleggsheaderen.



Figur 4: Eksempel på en rekke med headere.

Den nåværende IPv6 spesifikasjonen definerer 6 tilleggsheadere [5]:

- Hop-by-Hop opsjons header
- Routing header
- Fragment header
- Authentication header
- Encrypted security payload
- Destination opsjons header

Vi kan se at IPv6 Next Header feltet, kan inneholde enten typen av tilleggsheader eller protokoll typen til payloaden, f.eks. TCP eller UDP. Som en konsekvens av dette må ikke header typen være i konflikt med protokoll typen, og av den grunn blir de tildelt ut av det samme området som består av 256 tall [4]. De fleste typene er kjent for IPv4 og IPv6, selv om noen er noe annerledes.

3 Grunnleggende ATM

ATM (Asynchronous Transfer Mode) er et resultat av CCITT's (nå ITU-T) forsøk på å standardisere bredbånds ISDN på midten av 80-tallet. ATM grenset opprinnelig nært opp til de voksende Synchronous Digital Hierarchy (SDH) standardene, og var først utviklet for å gi kommunikasjons kanaler med vilkårlig båndbredde, innen et multipleksende hierarki bestående av et definert sett med kanaler av fast båndbredde. Grunnen til at ATM også kan gi kanaler med variabel bit rate, er en bivirkning som oppsto p.g.a. anskaffelsen av kanaler med vilkårlig kapasitet [26].

ATM var i begynnelsen en teknologi for telekommunikasjons fellesskapet, men på begynnelsen av 90-tallet ble ATM sett på, av data kommunikasjons fellesskapet, som en god kandidat i lokale nettverk, samt som en erstatning for TDM (Time Division Multipleksing) i transmisjonssystemer.

ATM har blitt en suksess som en link lags teknologi, fordi den tilbyr høyhastighets forbindelser til rutere og nettverk gjennom et fleksibelt, høyhastighets og skalerbart link lag.

Integrerte tjenester i Internet er blitt en realitet, med ATM som den viktige grunnstammen i nettet. "Klassisk" IP over ATM er nå vanlig og løser effektivt problemet med "best effort" tjenester i Internet med ATM linker [30]. Et viktig problem er å integrere ATM nettverk med Integrert Tjenester Internet. RSVP (Resource Reservation Protocol) er oppkobling eller signalerings delen av Integrerte Tjenester modellen i Internet.

En annen grunn til å benytte ATM er muligheten til å integrere eksisterende telefoni, video og data i LAN og WAN, siden den samme protokollen kan bli brukt for både LAN og WAN.

ATM Forum

ATM Forum er internasjonal organisasjon som ble grunnlagt i 1991, med mål å øke bruken av ATM produkter og tjenester gjennom en rask sammensmelting av interopererende spesifikasjoner. I tillegg fremmer ATM Forum, samarbeid og forståelse i industrien. I dag så har ATM Forum over 900 medlemsbedrifter, og den er åpen for alle organisasjoner, som interessert i ATM baserte løsninger.

Hensikten med ATM Forum er å utvikle spesifikasjoner vedrørende bruk av ATM. ATM's opprinnelse som en telekommunikasjonsstandard og dens raske inntog i LAN, MAN og WAN miljøer, krever av den må gjennom forskjellige standardiseringsmyndigheter (ITU-T, ANSI, IETF).

3.1 ATM terminologi

ATM – Asynchronous Transfer Mode – er et komplement til, og er utviklet fra Synchronous Transfer Mode.

Synchronous Transfer Mode er et synkront tidsmultiplekset system hvor pakker blir overført i definerte 125ms tidsluker. Synkron tidsmultipleksing gir brukeren full aksess til kanalen i et gitt tidsintervall. Dette betyr at tidsluken vil være tom, hvis brukeren ikke har noe å sende [3].

Asynchronous Transfer Mode er basert på asynkron tidsmultipleksing. Den grunnleggende teorien bak ATM, er at pakker som sendes merkes i forhold til forbindelsen, og blir ikke sendt ved en spesifisert tid. Asynkron tids multipleksing tillater at brukeren får vilkårlig aksess til kanalen når den har noe å sende, og den vil da få full aksess til kanalen under hele transmisjonen [3].

3.2 ATM celler

Den viktigste og mest signifikante delen av ATM er pakkene, som i ATM blir kalt celler fordi de har en fast lengde på 53 bytes. Cellene er små, noe som reflekterer opphavet i telekommunikasjonsnettverk. Headeren består av 5 bytes, og resten er reservert for data (48 bytes).

ATM cellene har ingen sender elle mottaker adresse fordi ATM er forbindelsesorientert og må ha satt opp en kanal mellom sender og mottaker før overføringen starter. For overføring benyttes kombinasjonen av forbindelsene VCI (Viritual Channel Identificator) og VPI (Viritual Path Identificator). Dette gjør svitsjingen raskere, fordi en gjør svitsjingen i henhold til 28 kontra 128 biter (IPv6) [3].

Det er to typer celler som er definert i ATM spesifikasjonen. En celle er definert for brukernettverk grensesnittet (UNI) og en for nettverk-nettverk grensesnittet (NNI), som vist i figur 5.

4	8	16	3	1	8	384 (48 bytes)
GFC	VPI	VCI	Type	CLP	HEC (CRC- 8)	Payload

12	16	3	1	8	384 (48 bytes)
VPI	VCI	Type	CLP	HEC (CRC- 8)	Payload

Figur 5: Spesifikasjon av cellene i ATM, der den øverste er en UNI celle og den nederste er en NNI celle.

- *GFC*: generic flow controll – 4 biter (bare for UNI)
- *VPI*: virtual path identifier – 8 biter for UNI og 12 biter for NNI
- *VCI*: virtual channel identifier – 16 biter
- *Type*: payload type – 3 biter
- *CLP*: cell loss priority – 1 bit
- *HEC*: header error correction – 8 biter
- *Payload*: 48 bytes med data

Generic Flow Control (GFC) vises ikke i celle headeren internt i nettverket, men bare i brukernettverk grensesnittet. Dette feltet kan hjelpe nettverket i å kontrollere trafikk flyten for forskjellige QoS [3,20].

Virtual path identifier (VPI) utgjør et ruting felt for nettverket. Den er 8 biter for UNI og 12 biter for NNI, og tillater at en kan ha flere virituelle baner i nettverket.

Den virtual channel identifier (VCI) blir brukt til ruting til og fra ende brukeren og fungerer som aksess punkt for tjenester.

Type (payload type) indikerer hvilken type informasjon som finnes i informasjons feltet. Hvis første bit er satt til 0, indikerer dette bruker informasjon.

Cell Loss Priority (CLP) feltet blir brukt til å guide nettverket når det oppstår metning. Hvis denne biten er satt til 0, indikerer dette at cellen har høy prioritet og bør ikke forkastes. Dette feltet er sentralt i analysen av QoS.

Hver ATM celle inneholder også et 8 biters header error correction (HEC) felt, som blir beregnet på bakgrunn av de gjenværende 32 bitene i headeren, og blir ikke bare brukt til feildeteksjon, men også i noen tilfeller feilkorreksjon.

VPI og VCI vil til sammen være den unike identifikatoren for en overføring eller forbindelse. Hovedparten av headerinformasjonen blir generert på ATM laget, men payload data blir generert av ATM adopsjons laget (AAL).

HEC blir generert av det fysiske laget, og ikke av ATM laget. HEC blir brukt til å detektere og/eller korrigere header informasjonen. En enkel feil kan bli detektert og korrigert, mens flere feil kan bare bli detektert.

3.2.1 UNI og NNI

Virtual Path (VP) forbindelser blir satt opp mellom to ende punkter, og kan enten være brukere, nettverksenheter eller en bruker og en nettverksenhet [3].

En *ende bruker til en ende bruker forbindelse*, blir brukt til å bære ende-til-ende data, men kan også bli brukt til å bære kontrollsignalering mellom ende brukere. En VPC mellom ende brukere gir dem en total kapasitet; VCC (Virtual Channel Connection) organiseringen av VPC (Virtual Path Connection) er opp til de to ende brukerene, gitt at settet med VCC'er ikke overskrider VPC kapasiteten [3].

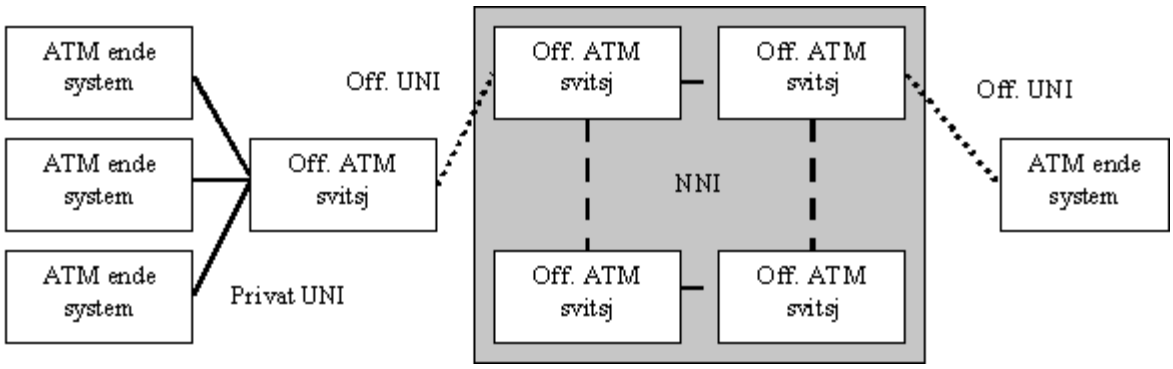
En *ende bruker til nettverks enhet forbindelse*, blir brukt til bruker-til-nettverk kontroll signalering. VPC kan bli brukt til å samle trafikk fra en ende bruker til en nettverks server.

En *nettverks enhet til nettverks enhet forbindelse*, blir brukt til nettverks trafikk håndtering og ruting funksjoner. En VPC kan blir brukt til å definere en vanlig rute for utveksling av nettverks håndterings informasjon.

UNI (bruker-nettverk grensesnittet) og NNI (nettverk-nettverk grensesnittet) er like. Forskjellen er at UNI vil forbinde bruker utstyr, f.eks. bredbånds terminaler, terminal adaptere, celle basert LAN/WAN utstyr og ATM svitsjer.

NNI kan bare forbinde porter i ATM svitsjer og behøver da ikke GFC feltet i ATM headeren. Det er tenkt å forbinde ATM subnettverk eller nettverk, og derfor har GFC feltet blitt innbefattet i VPI feltet og tillater dermed 16 ganger flere virtual paths.

Den private UNI blir brukt mellom ATM ende systemer i private nett og svitsjen som forbinder dem. Offentlige UNI blir brukt til å forbinde en komponent fra det private nettverket til det offentlige nettet. Se figur 6.



Figur 6: Forskjellen mellom private og offentlige UNI og NNI.

NNI blir brukt mellom svitsjene i det offentlige nett, og siden den eneste flyt kontrollen er den som tilbys av UNI, betyr dette at en svitsj er bare ansvarlig for å kontrollere bruker data, som har sitt opphav i en samme svitsjen. Med andre ord, er det svitsjen som må ta seg av flyt kontrollen mellom svitsjene.

3.3 ATM forbindelser

ATM forbindelser blir karakterisert som [20]:

- *Permanent Virtual Connections* (PVC)
  - Etablert gjennom en ekstern mekanisme, typisk en nettverk håndterings stasjon.
  - Alle svitsjer mellom sender og mottaker er programmert med de rette verdiene til VCI/VPI.
- *Switched Virtual Connections* (SVC)
  - Etablert av en applikasjon gjennom signalerings protokoller.
  - Mange høyere lags protokoller er basert på bruken av SVC'er.

og også som:

- *Punkt-til-punkt*
  - Kan være enten uni- eller bidireksjonal.
- *Punkt-til-flerpunkt*
  - Der en rot node forbindes til flere noder.
  - Celle replikasjon blir gjort av ATM svitsjen i hver grein.
  - Bare uni-direksjonal.

LAN teknologier tillater flerpunkt-til-flerpunkt forbindelser, og støtte for dette i ATM, ville forenklet LAN-LAN implementasjoner.

AAL5 (ATM Adaption Layer 5), som ofte blir brukt i data kommunikasjon, har ingen mulighet til å skille mellom celler som hører til andre pakker som mottas på den samme logiske forbindelsen. Dette betyr at alle cellene må komme i rekkefølge, for at pakken skal reassembleres [3].

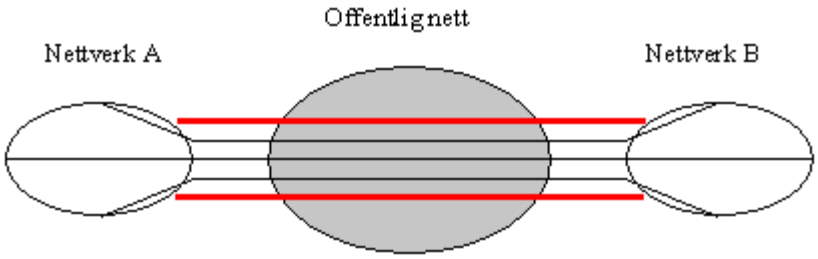
AAL3/4 har denne funksjonen i MID (Message Identifier) feltet, men det er ikke ønskelig å benytte denne protokollen, fordi den er mer kompleks enn AAL5 [3].

3.3.1 Virtual channels (VC) og virtual paths (VP)

ATM er forbindelses orientert og krever at en virtuell forbindelse blir etablere ved bruk av styring og automatiserte kall prosedyrer. All trafikk, som hører til den samme virtuelle forbindelsen, blir svitsjet den samme veien gjennom nettet. ATM forbindelser er av to typer, nemlig *virtual paths* (VP) og *virtual channels* (VC) [3].

VC'er er uni-direksjonale logiske ATM forbindelser med noen reserverte ressurser. V*irtual channel connections* (VCC) er den grunnleggende enheten for svitsjing i et ATM nettverk.

En VP er et antall virtuelle kanaler som har de samme ende punktene, og blir betraktet som en enhet for uni-direksjonal trafikk. Derfor blir alle cellene som går over alle VCC'er i en VPC svitsjet sammen. *Virtual paths* blir identifisert gjennom VPI indikatoren, og virtuelle kanaler gjennom kombinasjonen VPI/VCI.



Figur 7: Eksempel på en Virtual Path.

Denne VP teknikken hjelper å holde styr på kontroll kosten, ved å gruppere forbindelser som deler den samme banen gjennom nettverket, til en enkelt enhet (se figur 7).

Prosessen med å sette opp en VP forbindelse er avkoblet fra prosessen med å sette opp en individuell VC forbindelse [20]:

- VP kontroll mekanismene inkluderer kalkulering av ruter, allokere kapasitet og lagring av tilstands informasjon til forbindelsen.
- For å sette opp en virtuell kanal, må det først være en VP forbindelse til destinasjons noden med tilstrekkelig ledig kapasitet for å støtte den virtuelle kanalen, og med den passende tjenestekvaliteten. En virtuell kanal blir satt opp ved å lagre den nødvendige tilstands informasjonen (VC/VP mapping).

Når det har blitt satt opp en VPC, er det mulig for ende brukeren å forhandle om opprettelsen av en ny VCC.

3.3.2 Svisjing i ATM

En av fordelene med å bruke ATM, er at svitsjingen blir gjort i hardware siden cellene har den samme størrelsen. I teorien er svitsje funksjonen ganske enkel.

- En celle mottas over en link med kjente VPI/VCI.
- Svitsjen finner den utgående linken med den nye VPI/VCI ved å slå opp i en tabell
- Cellen blir så transmittert over linken.

Forskjellen fra Ethernet, er at svitsjene i ATM nettverket er involvert i forbindelses prosessen, ved å lage tilstander og å kvittere mellom svitsjer. Hvis prosessen er forstyrret på en eller annen måte, må andre svitsjer fjerne tilstandene. ATM standarden spesifiserer ingen ruting protokoll. ATM svitsjen fungerer som en VP eller VC svitsj for hver forbindelse.

3.4 Signalering i ATM

Signalering i ATM blir delt opp i tre lag:

- Lag 1: Signalering mellom fysiske hardware enheter.
- Lag 2: Sammenkobling av samarbeidende enheter.
- Lag 3: Etablering av samtaler og forbindelser.

For B-ISDN, blir UNI signalering lokalisert til lag 3. Signalering i ATM spenner over to forskjellige kontroll områder [20]:

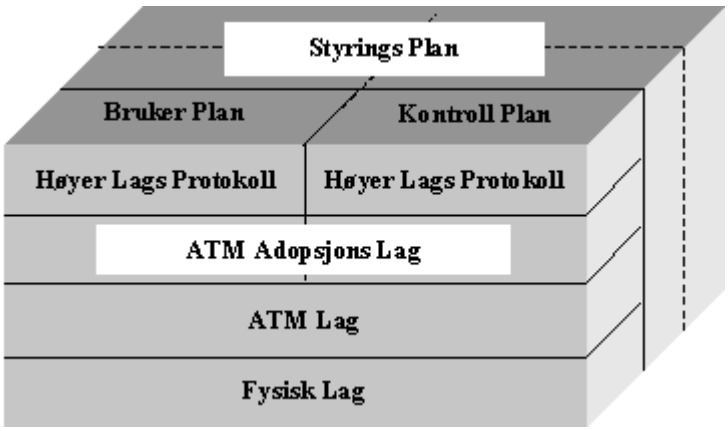
- *Forbindelses (Bærer) Kontroll*. Prosedyrer for å sette opp eller initialisere trekk av bruker data forbindelser, f.eks. ATM forbindelse, prosessen med å koble opp.
- *Samtale Kontroll*: Prosedyrer for å vedlikeholde selve forbindelsen, f.eks. forbinde spesifikke VPI'er eller VCI'er med bruker, rense VPI/VCI tabeller.

To viktige oppgaver eksisterer i alle signalerings scenarier [20]:

- *Nettverks avhengige oppgaver*
  - oppkobling, vedlikehold og fjerning av VCC'er og VPC'er.
  - forhandling av trafikk karakteristikker.
- *Tjeneste avhengige oppgaver*
  - uavhengig av alle spesifikke nettverks egenskaper.
  - ikke obligatorisk, men kan integreres.
  - definisjon og støtte av multicast og multipeer.
  - symmetrisk og asymmetrisk oppførsel til forbindelsene.
  - forhandling av QoS parametre.

Det siste punktet i *tjeneste avhengige oppgaver* impliserer at QoS er relatert til nettverket og tjenester gjennom trafikk styrings konseptene Peak Celle Rate (PCR), Minimum Celle Rate (MCR) og Vedvarende Celle Rate (VCR) og QoS attributtene Celle Taps Forhold (CTF), Celle Overførings Forsinkelse (COF), Celle Forsinkelses Variasjon (CFV) og Skur Toleranse (ST).

3.5 ATM referanse modellen



Figur 8: ATM referanse modell.

ATM referanse modellen har lag i to dimensjoner som kan ses i figur 8. De horisontale lagene tilsvarer de vanlige nettverkslagene fra OSI modellen. Altså det fysiske lag, ATM laget, ATM adopsjons laget og andre høyere nivåer. De vertikale lagene fungerer på tvers av alle de horisontale lagene, og er typisk opptatt av sammenhengen mellom de horisontale lagene. Altså bruker plan, kontroll plan og styring plan.

Bruker planet er opptatt av transmisjon av bruker data, og gir også tilknyttede kontroller som flyt kontroll og feil kontroll. *Kontroll planet* utfører samtale kontroll og forbindelses kontroll funksjoner. *Styrings planet* består av funksjoner som blir brukt til å påvirke og koordinere aktiviteter tilknyttet bruker- og kontroll planet. Planet inkluderer *plan styring*, som styrer funksjoner relatert til systemet som helhet og gir koordinering mellom planene, og *lag styring* som utfører styrings funksjoner relatert til resurser og parametere som oppholder seg i protokoll helheter [20].

3.5.1 Det fysiske laget

Det fysiske laget kontrollerer sending og mottak av biter over det fysiske mediet. Det holder også orden på grensene til ATM cellene, f.eks. hvor den celle starter og stopper. Det fysiske laget er ansvarlig for å sette inn celler i den riktige ramme typen for det fysiske mediet som blir brukt.

Det fysiske laget består av to sublag [20]:

- *Fysisk Medie Sublag* (PM) som er opptatt med transmisjon og mottak av en kontinuerlig strøm av biter med synkroniserings informasjon. Signalene blir konvertert til elektriske eller optiske signaler, og overført på optiske fibre, coax kabler eller radio linker.

- *Transmisjon Konvergens Sublag* (TC) som begrenser celle størrelsen, genererer og sjekker HEC, og setter inn og fjerner tomme ATM celler for transmisjons hastighets adopsjon.

3.5.2 ATM laget

ATM laget gir et grensesnitt mellom AAL og det fysiske laget. Dette laget er ansvarlig for å bringe celler fra AAL til det fysiske laget for transmisjon, og fra det fysiske laget til AAL til bruk i ende systemet [3].

ATM laget er hovedsakelig ansvarlig for generering av headeren til cellene og funksjonene som er tilknyttet headeren, d.v.s. svitsjing og ruting av celler, flyt kontroll, metnings kontroll, feil kontroll av bit i headeren og celle tegning.

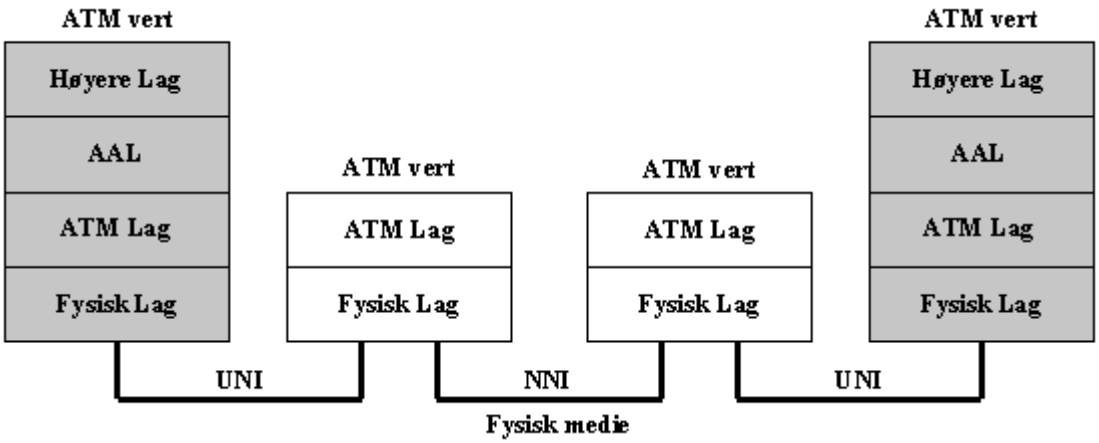
ATM laget mottar 48 oktetter med payload fra høyere lags trafikk, utfører generering av header og sender cellene til TC sublaget slik at rekkefølgen blir holdt innen den virtuelle forbindelsen.

På mottakersiden mottar en en strøm av celler, pakker ut headeren og leverer cellene, i rekkefølge, til den passende AAL tjeneste aksess punktet ved å bruke VCI/VPI verdiene som identifikatorer.

Ved svitsje elementer bruker ATM laget VCI/VPI til å rute celler. VPI og VCI verdiene kan forandre seg ved hvert svitsje element, og ATM laget gjør denne oversettelsen. Tolking av verdiene i payload typen (PT) og celle taps prioritet (CLP) feltet blir gjort ved ATM laget i svitsje elementene.

Med andre ord, ATM laget etablerer forbindelsen gjennom ATM nettverket, og svitsjer ATM cellene nodene basert på VPI/VCI verdiene.

Det fysiske laget og ATM laget, slått sammen, gir fasilitetene til den forbindelses orienterte transporten av celler. Disse to protokoll lagene må være til stede i hver ATM enhet, inkludert ende-systemet og bredbånds svitsjesystemer som vist i figur 9 under. Headeren til ATM cellene definerer funksjonaliteten til ATM laget siden den kan ha to former, UNI og NNI [20].



Figur 9: Protokoll stakken og sammenhengen mellom UNI og NNI.

3.5.3 ATM adopsjons lag

Hovedoppgaven til ATM adopsjons laget er å forsterke tjenesten til ATM laget til kravene til den spesifikke tjenesten. Dette inkluderer mapping av bruker/kontroll/styring PDU'er inn til ATM celle payloader og vice versa [3].

Når en tilpasser trafikken fra høyere lags protokoller til celle formatet, segmenterer ATM AAL trafikken inn i 48 byte deler. All tilpasning mellom forskjellige tjenester og transmisjon blir gjort i adopsjons laget. AAL er logisk delt inn i to sublag [3]:

- *Convergence Sublayer* (CS). Dette sublaget inneholder forskjellige funksjoner i henhold til AAL type som blir brukt. Den tilbyr funksjoner som trengs for å støtte spesielle applikasjoner som benytter AAL. Hver bruker forbindes til ALL laget gjennom et tjeneste aksess punkt – typisk adressen til applikasjonen. Dette gjør laget tjeneste avhengig.

- *Segmentation og Reassembly Sublayer* (SAR). Dette sublaget er ansvarlig for å pakke informasjons som mottas fra CS i celler for transmisjon, og å pakke ut informasjon i den andre enden.

Bruken av ATM gjør det nødvendig for adopsjonslaget å støtte informasjons transport protokoller som ikke er basert på ATM.

Tjenester som tilbys av AAL er typisk [3]:

- Håndtering av transmisjons feil.
- Segmentering og reassemblering.
- Håndtering av tapte og feilinskutte celler.
- Flyt og timing kontroll.
- Korreksjon av enkle bit feil i payloaden.
- Melding av tapte pakker og celler i feil rekkefølge.

For å minimalisere antall forskjellige AAL protokoller som må spesifiseres for å møte varierende behov, har CCITT definert fire tjeneste klasser som dekker en rekke krav.

3.5.3.1 ATM AAL tjeneste klasser

Klassifiseringen er basert på om et timing forhold må vedlikeholdes mellom kilde og destinasjon, om applikasjonen krever konstant bit rate og om overføringen er forbindelses orientert eller forbindelsesløs. De forskjellige typene trafikk og transmisjons tjenester som støttes i ATM blir delt inn i fire klasser [20]:

- *Klasse A* krav (f.eks. krets emulasjon):
  - en timing relasjon mellom kilde og destinasjons noden.
  - en konstant bit rate.
  - en forbindelses orientert overførings tjeneste.
- *Klasse B* krav (f.eks. variable bit rate video eller video konferanse):
  - en timing relasjon mellom kilde og destinasjons noden.
  - en variabel bit rate.

  - en forbindelses orientert overførings tjeneste.
- *Klasse C* krav (f.eks. data overførings applikasjoner):
  - ingen timing relasjon mellom kilde og destinasjons noden.
  - en variabel bit rate
  - en forbindelses orientert overførings tjeneste.
- *Klasse D* krav (f.eks. data overførigs applikasjoner):
  - ingen timing relasjon mellom kilde og destinasjons node.
  - en variabel bit rate.
  - en forbindelsesløs overførings tjeneste.

Disse tjeneste klassene har nå blitt forkastet, men fire typer tjeneste protokoller for AAL har blitt anbefalt av CCITT, basert på disse tjeneste klassene.

3.5.3.2 ATM adopsjons lags tjeneste protokoller

Først definerte CCITT (nå ITU-T) en protokoll type for hver tjeneste klasse, som fikk navnene Type 1 til Type 4. Hver protokoll består faktisk av to protokoller, en ved CS sublaget og en ved SAR sublaget. I det siste ble Type 3 og 4 slått sammen til Type ¾ og en ny type, Type 5, ble definert [20].

- **AAL1** tilsvarer tjenesteklasse A og er designet for å transportere CBR (constant bit rate) data strømmer på en slik måte at klokke informasjon kan gjennomrettes på mottaker siden. Eksempler på bruk er telefon og ukomprimert video. Payloaden består av samplet, synkron data. Sekvens nummer og sekvens nummer beskyttelse passer på at pakkene kommer fram i riktig rekkefølge. Denne protokollen krever et medium som kan overføre klokke.

- **AAL2** tilsvarer tjenesteklasse B og var tenkt til å transportere VBR (variable bit rate) data strømmer på samme måte som AAL1, men er enda ikke definert.

- **AAL3/4** tilsvarer tjenesteklassene C og D, og ble designet for å bære VBR strømmer uten eksplisitt timing informasjon. AAL3 er forbindelses orientert, og AAL4 er forbindelses løs, selvom det er uklart hvilken mening denne forskjellen har i ATM, og disse to blir vanligvis slått sammen.

- **AAL5 eller SEAL (Simple and Effective Adaption Layer)** svarer til tjeneste klasse C (ingen timing, VBR og forbindelses orientert), og ble utviklet p.g.a. oppfatningen av at AAL3/4 var ueffektiv.

4 Tjenestekvalitet (QoS)

Tjenestekvalitet eller Quality of Service (QoS) er en av de mest flyktige og forvirrende begrepene i datakommunikasjon i dag. Grunnen til dette kan være at det finnes så mange forskjellige definisjoner av emnet. Fagpressen, hardware og software bransjen, forbrukere og forskere ser ut til å ha sin egen definisjon av hva QoS er, hva QoS vil gjøre og hvordan en effektivt kan levere det. Dessverre vil disse definisjonene ofte resulterer i konflikter som ofte involverer komplekse, upraktiske og ukompatible mekanismer for å levere de ønskede resultatene. Hvis en skal kunne levere QoS, er en først nødt til å forstå og definere problemet. Hvis en analyserer begrepet QoS, det vil si at en går til hjertet av problemet, vil en sjelden finne en enkel definisjon av QoS. Selve begrepet QoS er et tvetydig begrep. Flere definisjoner av et begrep vil resultere i flere mulige løsninger.

4.1 Hva er Quality of Service?

For å kunne forstå begrepet QoS, må en først forstå ordene *quality* og *service*.

Quality kan beskrive mange egenskaper i datakommunikasjon, men generelt vil vi bruke *quality* for å beskrive prosessen med å levere data på en pålitelig måte, kanskje også bedre enn normalt. Denne definisjonen omfavner temaene tap av data, minimal eller ingen forsinkelse, konstante forsinkelseskarakteristikker (jitter) og muligheten til bestemme den mest effektive måten å utnytte nettverksressursene på (f.eks. den korteste veien mellom to kommuniserende noder). *Quality* kan også bety en spesiell egenskap, noe som gjør at *quality* også blir brukt til å definere spesielle karakteristikker ved bestemte nettverks applikasjoner eller protokoller [7].

Begrepet *service* kan også oppfattes på forskjellige måter, avhengig av hvordan en organisasjon eller foretak er strukturert. *Service* blir generelt brukt til å beskrive noe som blir tilbudt brukeren av et nettverk, som for eksempel ende-til-ende kommunikasjon eller klient-server applikasjoner. *Service* kan dekke et bredt spekter av tilbud, fra for eksempel elektronisk post til desktop video, Web tilgang og Chat rom. I et nettverk bestående av flere protokoller kan begrepet *service* ha flere betydninger. I et Novell NetWare nettverk, blir hver SAP annonse sett på som en individuell tjeneste eller *service*. I andre sammenhenger kan andre tjenester bli kategorisert etter forskjellige protokollrekkefølger, som for eksempel SNA, DECnet, AppleTalk og så videre. På denne måten kan en få en mer presis klassifisering av tjenester. Det er ikke så vanskelig å forestille seg en mer kompleks tjenesteklassifisering der du kanskje klassifiserer tjenester etter protokoll type (f.eks. IPX, DECnet), for så å klassifisere dem videre etter en mer presis protokoll sammenheng (f.eks. SAP typer innen IPX) [7].

Tradisjonelle tjenestetilbydere har brukt en rekke metoder for å kunne tilby *tjeneste garantier* til abonnenten sine, som for det meste er kontraktsbundet. For eksempel at nettverket er tilgjengelig er en av de mest vanlige målbare punktene som inngår i en avtale (SLA – Service Level Agreement) mellom tilbyder og abonnent [7]. Her er tilgang til nettverket den grunnleggende tjenesten, og hvis det på et gitt tidspunkt ikke kan tilbys nett-aksess er dette et brudd på avtalen. Hvis nettet ikke er aksesserbart er det grunn til å stille spørsmål rund kvaliteten (*quality*) på tjenesten. Av og til bruker tjenestetilbyderene tilleggs kriterier for å definere muligheten for å kunne tilby en *tjeneste garanti*, som for eksempel mengde trafikk til hver bruker. Hvis tjenestetilbyderen ikke klarer å levere mer en 98% av trafikken som er lovet, kan en muligens fastslå at tjenesten har dårlig kvalitet.

Forståelig vil en ikke benytte ordet *garanti* for et nettverkssegment, fordi uttrykket kan være tvetydig eller missledende. Dette spesielt i en omgivelse der praktisk talt all trafikk er pakkebasert og der tap av pakker ikke alltid er skadelig, men en måte å dynamisk tilpasse seg en datarate på (jf. TCP). Å tilby en tjenestegaranti impliserer ikke bare null pakketap, men også at ytelsen til nettverket er stabilt og forutsigbart. Dette vil by på en enorm utfordring, da flesteparten av dagens nettverk er pakkebasert [3].

Sammensettingen av ordene *quality* og *service* lager en ganske enkel definisjon, det vil si et mål på hvor godt nettverket oppfører seg og et forsøk på definere karakteristikker og egenskaper til spesielle tjenester. Likevel gir denne definisjonen rom for frihet og kreativ tolking, som blir reflektert gjennom forvirringen som er tilstede i nettverksbransjen. Et punkt går imidlertid igjen i de fleste definisjoner av *Quality of Service*. Det er muligheten for å differensiere mellom trafikk- eller tjenestetyper, slik at brukere kan behandle en eller flere klasser av trafikk forskjellig. Det er viktig å lage en forskjell på det som blir kalt tjenesteklasser (*Classes of Services - CoS*) og det mer tvetydige *Quality of Service*. Selvom noen mennesker vil kalle disse utrykkene som synonymer, har de en fin forskjell. QoS har et bredt og tvetydig innhold. CoS impliserer at tjenester kan bli kategorisert inn i separate klasser, som igjen kan bli behandlet forskjellig (se kapittel 4.5) [7].

Et annet viktig aspekt ved QoS er at, uavhengig av hvilke mekanismer som blir brukt, noe trafikk må bli gitt en forutsigbar tjeneste klasse. Med andre ord, det er ikke alltid viktig at noe trafikk blir gitt fortrinnsmessig behandling over andre typer trafikk, men isteden at karakteristikken til nettverket forblir forutsigbar [6]. Disse atferdskarakteristikkene er avhengig av flere spørsmål, som ende-til-ende responstid (bedre kjent som RTT – Round Trip Time), forsinkelse, køforsinkelse, tilgjengelig båndbredde, eller andre kriterier. Noen av disse karakteristikkene kan være mer forutsigbare enn andre, avhengig av applikasjonen, trafikk typen, kø- og bufferkarakteristikken til nettverksenhetene og arkitekturen til nettverket [3].

Det finnes to typer forsinkelse, nemlig *ekte* og *påført*. Med *ekte forsinkelse* mener vi den fysisk bindende karakteristikken til transportmediet og RTT til data, som er bundet til hastigheten til elektromagnetisk stråling. Dette blir kalt problematikken rundt lysets hastighet, fordi å overstige lysets hastighet ses på som umulig og at det er denne hastigheten som bestemmer hvor lang tid data minimum vil bruke over en link [7].

Påført forsinkelse er forsinkelsen som oppstår i nettverket i det pakker går inn i køen i forskjellige nettverksenheter, prosesseringsforsinkelse hos mottaker og metningen som er til stede mellom punkter i pakkens rute. Kø forsinkelse er ikke godt organisert i store nettverk. Variasjon i forsinkelsen over tid kan bare bli beskrevet som kaotisk, noe som gjør *påført forsinkelse* til den største usikkerheten i estimater av RTT på protokoll nivå [7].

Hvorfor er et nøyaktig estimat av RTT så viktig? For at en pakkesvitsjet protokoll skal kunne utnytte nettverket mest effektivt, skal protokollen sende en ny pakke ut på nettet idet en pakke fra den samme flyten blir mottatt korrekt. RTT blir bl.a. brukt til å styre TCP's timeout, retransmisjonsalgoritmer og algoritmer for dynamiske rate håndtering [3].

Så hva er *Quality of Service*? Det er foregått en viktig diskusjon på skaffe en mekanisme i nettverket som hindrer en abonnent fra å bruke alle tilgjengelige ressurser i nettverket når det er en kamp om ressursene. Eller å hindre en abonnent fra å bruke en stor andel ressurser slik at det forstyrrer en annen abonnent. Dette er en nobel tankegang, men en bør notere seg at det er en

direkte sammenheng mellom antall abonnenter og total tilgjengelig båndbredde, som kan uttrykkes som:

Tilgjengelig båndbredde / Antall brukere = Tenkt båndbredde pr. bruker

Likning 1

Hvis ikke nettverket er overdimensjonert for å takle alle mulige brukere uten metning i noen faser av nettverket, er en rettferdig tildeling av båndbredde pr. bruker, ikke et særlig troverdig system.

Hva som trengs for å oppnå QoS, eller mer viktig, differensiert CoS, er en måte å kunne tilby førsteklasses behandling av en klasse trafikk og "best effort" til en annen. Mange mekanismer forsøker å oppnå dette, og en må kanskje ta i bruk flere mekanismer sammen for å kunne oppnå noen grad av suksess. Noen av de mekanismene som tilbyr denne funksjonaliteten er [7]:

- **Trafikk forming.** Bruk av en "leaky bucket" for å planlegge trafikk inn i separate utkøer for å kunne gi en viss form for forutsigbarhet. Dette kan bli gjort på IP laget eller enda lenger ned ved ATM laget.

- **Adgangskontroll.** Fysisk begrense link hastigheten ved å klokke en krets til en viss datarate eller bruke en "token bucket" for å kontrollere innkommende trafikk. "Token bucket" implementasjonen kan være en frittstående implementasjon eller en del av arkitekturen til IETF Integrated Services.

- **IP prioritet.** Ved å bruke IP headeren sitt prioritet felt for å oppnå åtte forskjellige trafikk klasser.

- **Differensiell metnings kontroll.** En metningskontroll algoritme gir fordelaktig behandling til trafikk klasser i perioder når det oppstår metning.

4.2 QoS i et historisk perspektiv

Tidligere, når nettverksteknologien var ung, var konseptet med QoS fraværende. Det å få pakker fra et sted til et annet var første prioritet. Det å få frem pakker til destinasjonen uten feil, kan sees på som en primitiv form for QoS. Det vil si at trafikken var enten sendt og mottatt vellykket, eller ikke. Deler av de underliggende mekanismene for TCP/IP har utviklet seg til å bruke dette mønsteret mest effektivt. Faktisk har TCP blitt ganske elegant på den måten den takler tap av pakker. Når det oppstår tap av pakker reduserer den vindusstørrelsen og går inn i *slow start*, som er en mekanisme for å unngå metning. Nesten alle IP ruting protokoller bruker en timer for å detektere når en pakke går tapt eller når forbindelsen er nede mellom to kommuniserende noder (jf. kapittel 5.4).

Tidligere var metningskontroll og differensiering av tjenester på langt nær et så kritisk spørsmål som det er nå [7]. Hovedinteressen lå i simpelt hen i å få til en trafikkflyt, holde linken og ruting systemet stabilt. Det har ikke skjedd mye i denne oppfatningen, annet enn at de samme problemene har forsterket seg. Tjenestetilbyderene er fremdeles plaget med de samme problemene, og faktisk står de ovenfor mange flere spørsmål av kompleks politikk, skalering og stabilitet. Kun i de senere tider har interessen for QoS blitt en drivende kraft.

4.3 Hvorfor er QoS så interessant?

I den kommersielle Internet verdenen, kan QoS være et konkurransefortrinn for å kunne tilby en bedre tjeneste enn konkurrerende tjenestetilbydere. Mange tror at tjeneste tilbyderene er like, bortsett fra tjenestene som de tilbyr. Med hensyn på dette, ser mange QoS som hjelpemiddel for å få slutt på dette synet, samt at de oppnår en bedre tjeneste og ytterligere inntekt. Når det eksisterer mekanismer for å kunne tilby differensierte CoS nivåer, slik at en abonnents trafikk kan bli behandlet annerledes enn andre abonnenters, kan en også tilsynelatende utvikle forskjellige økonomiske modeller som en kan basere disse tjenestene på [7].

Ved å bruke samme logikk, finner man også like interessante grunner til å tilby differensiert CoS i bedriftsnett, universitetsnett og forskningsnett. Selvom de konkurransemessige og økonomiske drivkreftene ikke er til stede i disse nettverkene, er det likevel være ønskelig å kunne tilby flere nivåer av CoS.

4.4 Hvorfor har ikke QoS blitt innført?

Flesteparten av de største tjenestetilbyderene streber kun etter å kunne møte det økende behovet for båndbredde, og har historisk sett ikke konsentrert seg med å se på måter å kunne tilby differensiert CoS på en intelligent måte [7]. Det er to grunner til dette:

- For det første har redskapene har vært primitive og implementeringen av disse redskapene på høyhastighetsnett, har tradisjonelt hatt dårlig innvirkning på ytelsen. De komplekse algoritmene for kømanipulasjon og omgruppering av pakker, som implementerer købasert differensiert CoS, har først nylig blitt pålitelige i høyhastighets nettverk.

- For det andre har det ikke vært eksistert verktøy for å måle at en trafikklasse faktisk blir behandlet bedre enn andre trafikklasser. Hvis tjenestetilbyderene ikke kan bevise ovenfor abonnentene sine at en bestemt mengde av trafikken blir behandlet med en høyere prioritet, samt at abonnentene ikke kan verifisere disse nivåene av differensiering, er QoS implementasjonen verdiløs.

Det er også noen grunner til at QoS historisk sett ikke har blitt innført i bedrifts- og universitetsnett. En grunn kan være at båndbredde er relativt billig for disse nettene, noe som gjør det ganske enkelt å øke båndbredden ettersom nettverket kun strekker seg over et fåtall bygninger og etasjer. Hvis dette er tilfellet, og at verktøyene for trafikkhåndtering er primitive og påvirker ytelsen, er det noen ganger fordelaktig å øke båndbredden ved metningsproblemer. I en global målestokk, blir overdimensjonering sett på som en økonomisk urimelig luksus. Innen et vel definert område, kan overdimensjonering være et kost effektivt alternativ til QoS strukturer.

4.5 Differensiert Classes of Service (CoS)

Når de fleste mennesker refererer til QoS, refererer de vanligvis til *differensierte Classes of Service*. Det er *differensiert* som er nøkkelordet i dette utrykket, fordi før en kan skaffe en høyere tjenestekvalitet til en bestemt abonnent, applikasjon eller protokoll, må en klassifisere

trafikken inn i forskjellige klasser. Deretter må en bestemme en måte å behandle de forskjellige trafikklassene på, når trafikken går gjennom nettverket. Dette bringer frem flere viktige kriterier [7].

Når det blir utført en differensiering, blir dette gjort for å identifisere trafikk etter visse kriterier og klassifisere den innkommende trafikken inn i klasser. Denne klassifiseringen kan skje ved hjelp av klassifiseringsmekanismer ved inngangen til nettverket eller lenger inn i nettverkstopologien. Selve differensieringen kan bli gjort på mange forskjellige måter, men noen av de mest vanlige måtene å differensiere trafikk på er identifisere og klassifisere trafikk etter [7]:

- **Protokoll.** Nettverk og transportprotokoller som IP, TCP, UDP, IPX, o.s.v.
- **Kilde protokoll port.** Applikasjons spesifikke protokoller som f. eks. Telnet, avhengig av adressen til mottaker.
- **Destinasjon protokoll port.** Applikasjons spesifikke protokoller som f. eks. Telnet, avhengig av adressen til mottaker.
- **Senders adresse.** Protokoll spesifikk adresse til sender som indikerer trafikkens opphav.
- **Mottakers adresse.** Protokoll spesifikk adresse til mottaker som indikerer trafikkens destinasjon.
- **Grensesnittet mot kilden.** Grensesnittet som trafikken ble gitt til et utstyr på, d.v.s. adgangs grensesnittet.
- **Flyt.** En kombinasjon av kilde og destinasjonsadresse, samt kilde og destinasjons port.

Differensiering blir vanligvis utført som en metode til å identifisere trafikk, idet den kommer inn på nettverket eller for å sikre at trafikken er klassifisert riktig, slik at når trafikken entrer nettverket er den en gjenstand for en mekanisme som vil tvinge den til å tilpasse seg til den ønskede politikk. Denne håndhevingen som skjer ved inngangen blir kalt *aktiv håndheving* og kan bli gjort for å sikre at trafikken ikke har blitt vilkårlig kategorisert eller klassifisert før den kommer ut på nettverket [7].

Alternativet til aktiv håndheving er *passiv adgang*. Her blir trafikken akseptert som den er fra en nedstrøms abonnent og overført over nettverket. Selve atskillelsen blir her gjort mellom initiell differensiering gjennom bruk av selektiv håndheving og tillatelse til at differensiering skal kunne skje utenfor det administrative domenet til nettverket. Det er visse farer med å benytte en *passivt adgangs* tjeneste. For eksempel, kan nedstrøms abonnenter prøve å gi deres trafikk større prioritet enn berettiget, for så å bruke flere nettverksressurser. Merkingen av trafikken kan ha forskjellige former. Det kan være en RSVP reservasjonsforespørsel, IP prioritet for å indikere fortrinn eller et ATM CLP (cell loss priority) bit som indikerer høyere prioritet av trafikken ved metning. Vanskeligheten med håndheving av trafikken ved endene av nettet, er avhengig av hvordan og ved hvilket lag i protokollstakken det blir gjort [7].

Basert på de foregående kriterier, kan ende-systemet ta flere handlingsalternativer etter at identifikasjon og klassifisering av trafikken er gjort. Et av de enkleste handlingsalternativene er å køe

klassene forskjellig, for å skaffe forskjellige tjenesteklasser. Flere andre valg er tilgjengelige, som f.eks. selektiv forwarding av forskjellige klasser langs forskjellige ruter i nettet. En kan gjøre dette med tradisjonell pakke-forwardingssystemer eller ved å mappe trafikklasser til spesifikke lag 2 svitsjede baner i nettverket [7]. En variasjon av å mappe trafikklasser til flere lag 2 svitsjede baner, er også å benytte forskjellige trafikkutformings systemer eller metningsterskler for hver virtuelle bane. Det finnes sikkert flere måter å skaffe differensierte tjenester på enn det som er beskrevet her, men det viktige her er at identifikasjon, klassifisering og merking av trafikk er det fundamentale konseptet for å kunne tilby differensiert CoS.

5 Hvordan kan IP versjon 6 garanter QoS?

Som nevnt tidligere gir IPv6 en rekke forsterkninger fra den nåværende IPv4 spesifikasjonen. Imidlertid er ikke integrert tjenestekvalitet en av argumentene som blir brukt for å få bedrifter til å implementere IPv6 i sine nett. Det er en vanlig misoppfatning at IPv6 protokollen inneholder en metode som garanterer tjenestekvalitet. Selvom strukturen til IPv6 protokollen er ganske forskjellig fra IPv4, er de grunnleggende funksjonelle kreftene ganske like.

To signifikante komponenter i IPv6 protokollen kan faktisk gi en metode for å lever differensierte Classes of Services (CoS). Den ene komponenten er et 8 biters Traffic Class felt i IPv6 headeren, mens den andre er en Flow Label [7].

5.1 Traffic Class feltet

Det 8 biters Traffic Class feltet i IPv6 headeren er tilgjengelig for sendende noder og/eller forwardende noder for å identifisere og skjelne mellom forskjellige klasser eller prioriteter av IPv6 pakker. Når dette dokumentet blir skrevet, er det en rekke eksperimenter på gang når det gjelder bruk av IPv4 sitt Type of Service og/eller prioritets biter for å kunne skaffe forskjellige former for differensierte tjenester for IP pakker, annet enn gjennom bruk av eksplisitt flyt oppsett. Traffic Class feltet i IPv6 headeren er tenkt brukt til liknende funksjonalitet i IPv6. Det blir håpet at disse eksperimentene vil før eller senere resultere i en enighet om hvilken typer trafikk klassifisering som er best for IP pakker [4,23].

De følgende generelle forutsetninger kreves av Traffic Class feltet [5,23]:

- Tjeneste grensesnittet til IPv6 tjenesten, innen en node, må sørge for at en høyere lags protokoll gir verdien til Traffic Class bitene i pakker som originerer fra høyere lags protokoll. Standard verdien må være null for alle åtte bitene.

- Noder som støtter en spesifikk bruk av noen eller alle Traffic Class bitene, har lov til å forandre verdien på disse bitene i pakker som de originerer, forwarder eller mottar, som krevd for den spesielle bruken. Noder skal ignorere og la de bitene som er angitt i Traffic Class feltet være, hvis noden ikke angir en bestemt bruk av de.

- En høyere lags protokoll må ikke anta at verdien av Traffic Class bitene i en mottatt pakke er den samme verdien som ble sendt fra kilden.

5.2 Flow Label feltet

Det 20 biters Flow Label feltet i IPv6 headeren kan bli brukt av en kilde til å merke sekvenser med pakker, kalt flyt, der det kreves spesialbehandling av IPv6 ruterene, som f.eks. ikke standard QoS eller real-time tjenester. Denne delen av IPv6 er foreløpig på eksperimentelt nivå og gjenstand for forandring [4].

Egenskapene til denne spesielle håndteringen kan bli gitt til ruterene av en kontroll protokoll, som f.eks. RSVP.

Det kan være mange aktive flyter fra en sender til en mottaker, pluss trafikk som ikke forbindes med noen flyt. En flyt blir unikt definert av kombinasjonen kilde adresse og flyt feltet. Pakker som ikke hører til en bestemt flyt, har flyt feltet satt til null.

Merking av flyten, blir gjort av noden som sender. Flytmerker blir valgt tilfeldig og uniformt fra området 1 til FFFFF hexadesimalt. Hensikten med denne tilfeldig allokeringen, er å gjøre enhver kombinasjon av biter innen flyt feltet egnet som en tilfeldig nøkkel, som ruterene kan bruke til å slå opp tilstanden til flyten [5].

Alle pakker som hører til den samme flyten må sendes med den samme kilde og mottaker adressen, samt flytmerket [4].

Den maksimale levetiden til en flyttilstand som blir etablert langs banen, må bli spesifisert i beskrivelsen av etableringsmekanismen for tilstanden, f.eks. RSVP. En kilde kan ikke bruke det samme flytmerket om igjen, innen levetiden til flyttilstanden.

Når en node stopper opp og starter igjen (f.eks. på grunn av krasj), må den være forsiktig med å ikke bruke et flyt merke som den har brukt tidligere. Dette kan oppnås ved å lagre bruken av flyt merker på en stabil plass som overlever krasj.

Det er ingen krav til at alle, eller flesteparten, av pakkene skal høre til en bestemt flyt.

5.3 Resource reservation protocol (RSVP)

Oppgaven til reservasjons protokoller er å opprette og vedlikeholde reservasjon av ressurser over en bane i et nettverk. Slike protokoller er signalerings protokoller, og ikke ruting protokoller. En annen misoppfatning er at protokollene i seg selv forbedrer tjenestekvaliteten. Den riktige beskrivelsen er at de gir en bedre tjenestekvalitet, av de nødvendige mekanismene som er til stede [22].

Tidligere reservasjons protokoller inneholder [20]:

- ST (Stream Protokoll – også kalt IPv5), som gir 1:1 duplekse reservasjoner
- ST-II, som gir 1:n simplekse reservasjoner, er sender orientert og bruker såkalte harde tilstander.

Nå er det Resourse Reservation Protocol (RSVP) som tiltrekker seg størst interesse. RSVP utvikles av RSVP arbeidsgruppen i IETF, og er nå en foreslått standard. RSVP protokollen blir brukt av verten for å spørre nettverket om en bestemt tjenestekvalitet for data strømmer eller flyter fra bestemte applikasjoner [22].

RSVP blir også brukt av ruterene for å levere QoS forespørsler til alle nodene langs trafikkflyten, og til å etablere og vedlikeholde tilstander for å kunne gi den etterspurte tjenesten. RSVP forespørsler vil generelt resultere i at ressurser blir reservert i hver node langs banen.

RSVP spør etter ressurser for simplekse flyter, d.v.s. at den spør etter resurser i bare en retning. RSVP opererer over IPv4 eller IPv6, og opptar plassen til en transport protokoll i protokoll

stakken. Protokollen transporterer imidlertid ikke data fra applikasjoner, men er snarere en Internet kontroll protokoll, som f.eks. ICMP, IGMP eller ruting protokoller [22].

En RSVP prosess spør den lokale ruting databasen, for å få forskjellige ruter. Et multicast tilfelle, kan f.eks. en vert sende IGMP meldinger for å bli medlem av en multicast gruppe, og sender så RSVP meldinger for å reservere ressurser langs banen til den gruppen. Ruting protokoller fastslår hvor pakker skal forwardes, mens RSVP bryr seg bare om tjenestekvaliteten til de pakkene som skal leveres [22].

For å effektivt kunne tilpasse seg store grupper, dynamisk medlemskap av grupper og forskjellige mottaker krav, gjør RSVP det slik at det er opp til mottakeren å kreve en bestemt QoS. En QoS forespørsel fra mottakeren bli levert til den lokale RSVP prosessen. RSVP protokollen sender denne forespørselen til alle nodene langs den reverse banen [20].

Tjenestekvalitet blir implementert for en bestemt data flyt, av mekanismer som samlet kalles *trafikk kontroll*. Disse mekanismene inneholder:

- Klassifisering av pakker.
- Adgangs kontroll.
- Planlegging av pakker, d.v.s. når pakker skal sendes videre.

Klassifiseringen av pakkene, bestemmer QoS klassen for hver pakke. For hvert utgående grensesnitt, gir pakkeplanleggeren den lovede tjenestekvaliteten. Trafikk kontroll implementerer QoS tjeneste modeller, som er definert av IETF sin arbeidsgrupper for integrerte tjenester.

Under oppsettet av reservasjonene, blir RSVP sin QoS forespørsel sendt til to lokale beslutnings moduler, nemlig *adgangs kontroll* og *overvåknings kontroll*. Adgangs kontroll modulen bestemmer om det er nok ressurser, slik at den tjenestekvaliteten som er etterspurt, kan gis. Overvåknings kontroll modulen bestemmer om brukeren har administrative rettigheter til å gjøre reservasjonen. Hvis begge undersøkelsene gir positivt resultat, blir parametere satt i pakke klassifisereren og i link lags grensesnittet, for å gi den ønskede tjenestekvaliteten [22].

Det som skiller denne protokollen fra andre er:

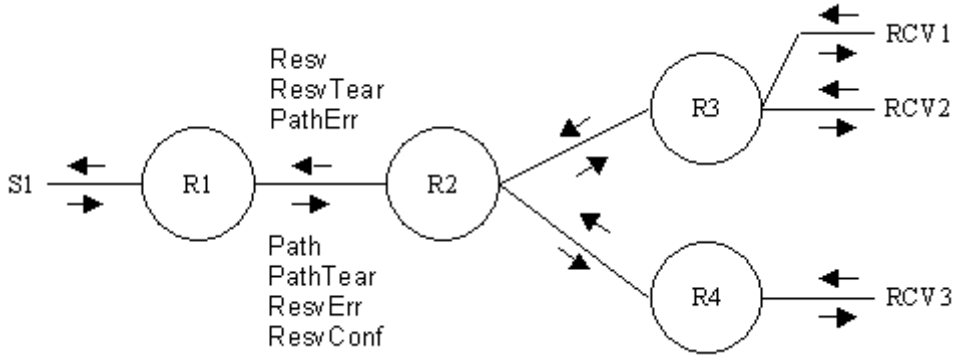
- *Mottaker orientert*. Selve reservasjonen bli initiert av mottakerene ved bruk av Reverse Path Forwarding, og ikke sendere, som tillater heterogene mottakere.
- *Reservasjons typer*. Sammensmelting av reservasjons forespørsler gjennom filtre, noe som gjør rutingen mer effektiv.
- *Myke tilstander*. Periodiske oppdateringer av tilstander, som tillater fleksibilitet i ruting og avpassing, men med fare for implosjon.

Videre gir RSVP m:n simplekse forbindelser, og på den måten støtter den både multicast og unicast reservasjoner. Protokoll meldingene blir enten transportert direkte over IP, ved bruk av protokoll nummer 46, eller innkapslet i UDP.

5.3.1 RSVP tilstander og meldinger

Figur 10 viser forskjellige meldinger som er tilgjengelige i RSVP og den resulterende meldings flyten, både oppstrøms (←) og nedstøms (→). Senderene vil varsle dens trafikk ved bruk av

Path meldingen, som mottakerene vil svare på med en Resv melding. Resv meldingen inneholder reservasjons forespørselen [20].



Figur 10: Meldingsflyt i RSVP.

I hver node i banen mellom sender og mottaker, blir det holdt to typer *soft states*, med følgende tilstands attributter:

- Path (fra sender til mottaker), som inneholder innkommende link og utgående linker.
- Reservation (fra mottaker), som inneholder, for hver utgående link, sender informasjon, beskrivelse av ressurs og reservasjons type.

Hver RSVP vert sender Path meldinger nedstrøms langs uni-/multicast banen som er gitt av ruting protokollen. Disse Path meldingene lagrer ”bane tilstanden” i hver node på veien. Denne bane tilstanden inneholder i hverfall unicast IP adressen til noden foran, som blir brukt til å rute Resv meldingen hopp-for-hopp i den reverse retningen [22].

En Path melding inneholder følgende informasjon i tillegg til adressen fra forrige hopp:

- Sender Template
  En Path melding må inneholde en Sender Template, som beskriver formatet til datapakkene som senderen generere.

- Sender Tspec
  En Path melding må inneholde en Sender Tspec, som definerer trafikk karakteristikken til data flyten, som senderen vil generere. Tspec blir brukt av trafikk kontrollen, for å hindre over-reservasjon og unødvendige feil i adgangs kontrollen.

- Adspec
  En Path melding kan inneholde OPWA averterings informasjon (omtales i kapittel 5.3.2), som kalles Adspec. En Adspec som blir mottatt i en Path melding, blir endt til den lokale trafikk kontrollen, som returnerer en oppdatert Adspec. Den oppdaterte versjonen blir sendt videre i Path meldingen som sendes nedstrøms.

Hver mottaker sender RSVP reservasjonsforespørsler (Resv) meldinger oppstrøms mot senderen. Disse meldingene må følge den eksakt reverse banen som data pakkene vil benytte. De kreerer og vedlikeholder reservasjonstilstanden i hver node langs banen. Resv meldingene må til sist komme fram til senderen, slik at senderen kan sette de passende trafikk parameterene for det første hoppet.

RSVP benytter *soft states* for å administrere reservasjonstilstandene i rutere og verter. RSVP sine *soft states* blir kreert og periodisk oppdatert av Path og Resv meldinger. Tilstanden blir slettet hvis ingen oppdaterings melding ankommer før timeout går (typisk hvert 30 sekund).

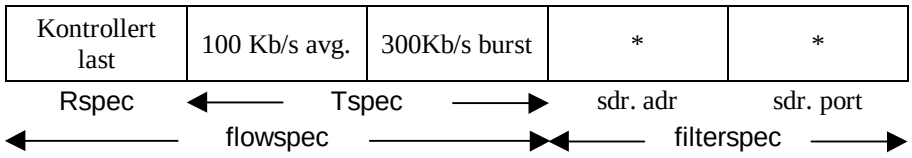
I tillegg, funksjonaliteten til de andre RSVP meldingene [20]:

- ResvTear, PathTear: eksplisitt sletting av myke tilstander.
- ResvErr, PathErr: Feil når en etablerer myke tilstander.
- ResvConf: Bekreftelse på en vellykket reservasjon.

5.3.2 RSVP reservasjon

Reservasjonen blir gjort for hver enkelt flyt, hvor hver flyt f.eks. identifiseres av IPv6 sin Flow Label.

Et detaljert eksempel på en reservasjons spesifikasjon (Resv melding) blir gitt i figur 11.



Figur 11: Eksempel på en Resv melding.

En RSVP reservasjons forespørsel består av en flowspec sammen med en filterspec. Til sammen kalles disse *flyt deskriptoren*. Flowspec spesifiserer den ønskede tjenestekvaliteten.
Filterspec definerer, sammen med sesjonsspesifikasjonen, den flyten som skal motta den tjenestekvaliteten, som er definert av flowspec. Flowspec blir til å sette parametere i nodens pakke planlegger, mens filterspec blir brukt til å sette parametere i pakke klassifisereren. Data pakker som er adressert til en bestemt sesjon, men som ikke passer til noen av filterspec'ene for den sesjonen, bli behandlet som "best effort" trafikk [20].

Flowspec i en reservasjonsforespørsel vil generelt inneholder en tjenesteklasse og to numeriske parametre:

- Rspec, som definerer den ønskede tjenestekvaliteten.
- Tspec, som beskriver data flyten.

Ved hver mellomliggende node, trigger reservasjons forespørselen to handlinger [22]:

1. *Gjøre en reservasjon på linken*. RSVP prosessen sender forespørselen til adgangs- og overvåknings kontrollen. Hvis forespørselen ikke kommer gjennom en eller begge kontrollene, blir reservasjonen avvist og RSVP prosessen sender en feilmelding tilbake til senderen av forespørselen. Hvis begge kontrollene var vellykket, setter noden pakkeklassifisereren til å velge de data pakkene som er definert av filterspec, og den samarbeider med link laget for å få den ønskede tjenestekvaliteten som er definert av flowspec.

De detaljerte reglene for å tilfredsstille en RSVP QoS forespørsel er avhengig av link lags teknologien som blir brukt ved hvert grensesnitt. For en enkel leid linje, vil en få den ønskede tjenestekvaliteten fra pakkeplanleggeren i link lags driveren. Hvis link lags teknologien implementerer en egen mulighet fot QoS håndtering, må RSVP forhandle med link laget, for å den ønskede tjenestekvaliteten.

2. *Videresending av forespørselen oppstrøms*. En reservasjons forespørsel blir sendt oppstrøms mot senderene. Reservasjons forespørselen som noden sender oppstrøms, kan være forskjellig fra forespørselen den mottok nedstrøms, av to grunner.

   - Trafikk kontroll mekanismen kan modifisere flowspec hopp-for-hopp.
   - Reservasjoner fra forskjellige nedstrøms greiner i multicast treet, fra den samme senderen, må smeltes sammen når reservasjonen er på vei oppstrøms.

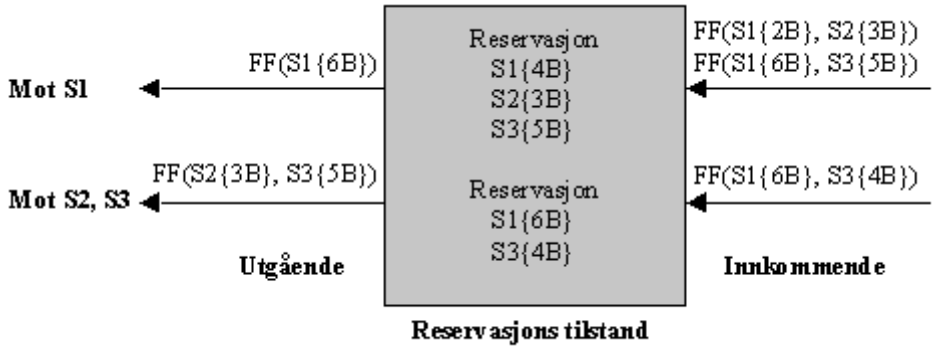
Hvis en Path melding består av en Adspec, som inneholder utvidede trafikk karakteristikker som f.eks. tjenesteklasse og som oppdateres i de mellomliggende ruterene, er det lettere å bestemme reservasjons forespørselen. En slik reservasjonsmodell kalles OPWA – One Pass With Adevertisement, og er den mest vanlige måten å gjøre reservasjoner i RSVP på i dag [22].

5.3.3 Tilstands oppdatering i RSVP

Det er tre mulige metoder som en definert for å smelte sammen reservasjoner, når en skal oppdatere reservasjonene i nodene [22].

- *Fixed filter* (FF), hvor senderene bli definert eksplisitt, og det er en separat reservasjon for hver sender.
- *Shared Explicit* (SE), hvor senderene blir identifisert eksplisitt, men reservasjonen er delt.
- *Wildcard filter* (WF), hvor senderene ikke er definert og reservasjonen er delt.

Shared Explicit og Wildcard filteret er egnet for situasjoner med flere sendere og mottakere, men med bare en aktiv sender av gangen (f.eks. videokonferanse). Fixed filteret er mer egnet for en enkelt sender med mange mottakere (f.eks. Video on Demand).



Figur 12: Oppdatering av reservasjonstilstander.

Et eksempel på en tilstandsoppdatering som bruker Fixed filter er gitt i figur 12. I dette eksempelet blir de innkommende reservasjons forespørslene smeltet sammen, ved å ta den største

forespørselen på det grensesnittet for den bestemte brukeren. De utgående forespørslene blir gitt av maksimumet av alle reservasjonene som er installert i den ruteren for den bestemte brukeren.

5.3.4 Problem områder i RSVP

Bruken av RSVP er pr. i dag fortsatt på et eksperiment nivå. En av de største bekymringene i dag, er at RSVP blir brukt i flere situasjoner enn det som var ment.

RSVP ble utviklet for real-time multimedia applikasjoner med en viss varighet og få sesjoner, slik som video konferanse (1Mbits/s pr. sesjon) [20]. I dag blir også protokollen brukt til å prioritere trafikk karakterisert av mange sesjoner med kort levetid, som f.eks. IP telefoni og surfing på web'en (http sesjoner). Resultatet av dette blir mange reservasjoner, store tilstands-tabeller og redusert ytelse i ruterene.

For å rette på dette, har det av IETF blitt utgitt noen retningslinjer [25]. Dette dokumentet peker også på et antall uløste problemer som berører RSVP:

- *Skalerbarhet*. Involverer tilstands prosessering i ruterene og krav til lagrings resurser for et stort antall sesjoner.
- *Sikkerhet*. Behandling av falske reservasjons forespørsler.
- *Overvåkning*. Adresserings mekanismer for hvem som kan, og ikke kan, lage reservasjoner.

Et annet fundamentalt problem er for hvilke typer applikasjoner som det skal være mulig å lage reservasjoner for. De nevnte punktene forskes det intenst på i dag.

5.4 Transmission controll protocol (TCP)

Det finnes mange forskjellige måter å skaffe differensierte tjenesteklasser (CoS), og hvor i nettverkstopologien dette kan gis.

Det stedet hvor det er mest passende å gi differensiering, er innen fellesnevneren, der fellesnevneren er definert som ende-til-ende plattformen i dagens nettverk. Med dette blir det et spørsmål om hvem komponent som har størst utbredelse i ende-til-ende trafikk banen [7].

I dagens Internet, er det TCP/IP protokollen som er den udiskutable fellesnevneren. Denne påstanden har flere årsaker. Fellesnevneren blir valgt med håp om å benytte den mest benyttede protokollen i nett, enten det er lag 2 eller lag 3. Ved å benytte den mest vanlig protokollen blir implementasjon, administrasjon og feilsøking mye enklere, og gir dermed en større sannsynlighet for å kunne lykkes med QoS implementasjonen [3].

Det er også tilfellet at denne teknologien opererer på en ende-til-ende basis, og benytter signaleringsmekanismer som spenner over hele nettverket på en gjennomført måte. IP, som TCP/IP ofte refereres som, er den ende-til-ende transport tjenesten som finnes i de fleste tilfeller [3]. Selv om det er mulig å lage QoS tjenester i lavere lag av protokoll stakken, dekker slike tjenester kun deler av ende-til-ende banen. Slike mekanismer blir ofte maskert bort av forvrengning

av den resterende ende-til-ende banen, der mekanismene er fraværende. En vil ofte se at utfallet av en slik QoS implementasjon ikke er effektiv.

Differensierte tjenesteklasser i IP laget, er ikke så komplekst som en kan tenke seg. Selve forvirringen er misoppfatningen av at en kan gi en teknisk garanti for at trafikken når mottaker, og det enda bedre enn "best effort" trafikk. Det er viktig at forstå at TCP/IP jobber i et forbindelsesløst, datagram orientert miljø og at det er ingen funksjonelle garantier for levering [3]. I steden for å gi et funksjonelt nivå for ytelse, arbeider TCP/IP etter en annen filosofi. TCP/IP prøver dynamisk ut nivået av tilgjengelige ressurser til sendere, nettverket og mottaker og forsøker å optimalisere bruken av ressurser slik at pakkene kan leveres på en pålitelig måte. Det er også viktig å forstå at TCP er designet for å takle tap av pakker på en stilfull måte [3].

TCP blir sett på som *selv-klokkende*, d.v.s. at når TCP fastslår at en pakke er tapt, slutter senderen å originere pakker, senker vindusstørrelsen og retransmitterer den tapte pakken med en lavere rate for avslutte den påbegynte transmisjonen. Etter at det har blitt etablert en pålitelig transmisjonsrate, begynner TCP å prøve ut om det er blitt flere tilgjengelige ressurser ved å øke transmisjonsraten helt til det ikke er flere tilgjengelige ressurser igjen. Denne prosessen kalles TCP *ramp up* og kommer i to utgaver [3]:

- *Slow-start*, som dobler vindusstørrelsen for transmisjon for rask å øke transmisjonsraten.
- *Congestion-avoidance*, som lineært øker vindusstørrelsen for å unngå metning.

Pakketap skjer når det maksimale nivået nås, og når dette skjer slutter senderen å sende igjen. Det er av denne grunn TCP kalles selv-klokkende.

5.5 Real-time transport protocol (RTP)

RTP er blitt designet innen IETF, og gir ende-til-ende transport funksjoner i nettverket, som er passer for applikasjoner som overfører real-time data, som f.eks. tale og video. Disse tjenestene inneholder identifikasjon av payload type, sekvens nummerering, tidsmerking og monitorering av levering. Applikasjoner har typisk RTP over UDP, for å dra nytte av dens multipleksing- og checksumtjenester.

RTP i seg selv, gir ingen mekanisme for å sikre riktig levering eller gi andre former for QoS garantier, men er avhengig av lavere lags protokoller til denne jobben [32]. RTP garanterer ikke at pakken kommer fram eller i riktig rekkefølge og den antar heller ikke at det underliggende nettverk gjør det heller. Sekvensnummerene i RTP, tillater mottakeren å rekonstruere sekvensen av pakker. Sekvensnummerene kan også bli brukt til å bestemme den passende lokasjonen til pakken, f.eks. i dekoding av video, uten nødvendigvis å dekode pakkene i sekvens.

Ettersom RTP hovedsakelig er designet for å tilfredsstille behovene til multimedia konferanser, er den ikke begrenset til bare denne applikasjonen. Lagring av kontinuerlig data, samt kontroll og målingsapplikasjoner finne RTP anvendelig.

RTP består av to tett tilknyttede deler [32]:

- *Real-time transport protokoll*, som bærer data som har real-time egenskaper.

- *RTP kontroll protokoll (RTCP),* som overvåker QoS og bringer informasjon om deltakerene i en pågående sesjon. Det siste aspektet til RTCP kan være nok for ”løst kontrollerte” sesjoner f.eks. hvor det ikke er noen eksplisitt medlemskapskontroll og set-up, men det er ikke nødvendigvis ment å støtte alle kontroll kommunikasjon krav i en applikasjon. Denne funksjonaliteten kan bli fullt eller delvis underordnet av en separat sesjons kontroll protokoll.

På grunn av at RTP ikke kan garantere noen form for QoS, vil jeg ikke gå noe videre inn på denne protokollen i dette dokumentet.

6 Hvordan kan ATM garantere QoS?

ATM teknologien er unik på den måten den, på et nettverk, støtter flere applikasjoner som krever forskjellig QoS. For å kunne garanter QoS og samtidig bruke de tilgjengelige ressursene effektivt, må nettverket støtte en rekke muligheter for trafikkhåndtering.

6.1 Trafikk kontrakten

Det økende behovet for effektiv kommunikasjon har ført til at det har blitt laget mange applikasjoner. Internet, World Wide Web browsing, telemedisin og telekonferanse er bare noen applikasjoner som er under sterk utvikling. Disse applikasjonene har varierende behov for båndbredde, trafikkarakteristikk og ende-til-ende ytelse. Ende-til-ende utgjør alle elementer av nettverket, mellom kilden og mottakeren [26].

Tidligere generasjoner av nettverk var utviklet for å håndtere forskjellige typer applikasjoner ved å bruke separate infrastrukturer. Hvert nettverk blir administrert separat og bruker forskjellig svitsjeutstyr og transmisjonslinker. Viktig infrastruktur, administrasjon og faste eiendomskostnader kan bli spart ved å samle alle tjenestene som tilbys til ett nettverk (jf Statoils visjon). ATM gir muligheten for implementering av et slikt samlet nettverk, siden det samtidig kan støtte den tjenestekvaliteten som er påkrevd av eksisterende og framtidige applikasjoner gjennom valg av en passende tjenestekategori.

En applikasjon som bruker et ATM nettverk, overbringer dens båndbredde og ytelses behov gjennom trafikk kontrakten. Trafikk kontrakten inneholder følgende komponenter:

- Tjeneste kategorien
- Den tilstrekkelige tjenestekvaliteten
- Trafikk karakteristikkene til forbindelsen
- En definisjon av hvordan trafikken skal oppføre seg

6.1.1 Tjeneste kategorier

For å effektivt å kunne muliggjøre dette multi-applikasjons miljøet i et nettverk, som potensielt består av utstyr fra flere leverandører, har det på ATM laget blitt definert og standardisert forskjellige tjenestekategorier. Disse ATM tjenestekategoriene er definert for å dekke det spektrum av eksisterende og framtidige applikasjoner, som benytter ATM nettverk. Tjenestene tilbyr forskjellige QoS forpliktelser, som forsinkelse- og tapstoleranse. Tjenestene er også forskjellig når det gjelder hvordan nettverket allokerer båndbredde og hvordan det bruker forskjellige håndteringsfunksjoner for trafikk [29]. Tjeneste kategoriene er:

- *Constant bit rate* (CBR) tjeneste
- *Variable bit rate* (VBR) tjeneste
- *Available bit rate* (ABR) tjeneste
- *Garanteed frame rate* (GFR) tjeneste
- *Unspecified bit rate* (UBR) tjeneste

For CBR og VBR tjenestene, blir båndbredden allokert av Connection Admission Control (CAC) gjennom varigheten til forbindelsen, selvom forbindelsen ikke benytter den hele tiden [20].

For å kunne bygge kosteffektive nettverk, er det et behov for å maksimalt kunne utnytte båndbredden. Båndbredden blir dynamisk tilgjengelig alt ettersom forbindelser blir satt på tomgang [26]. ABR, GFR og UBR tjenestene benytter den dynamisk tilgjengelige båndbredden, men for tjenestene ABR og GFR kan det statistisk bli allokert en minimum båndbredde.
Disse tjenestene kalles *bandwith on demand* (BoD) tjenester. Målet med BoD er å, så fort som mulig, få en del av den båndbredden som blir dynamisk tilgjengelig i nettverket til eget forbruk. Forskjellig fra UBR, kan ABR og GFR tjenestene garantere at nettverket bærer en minimum båndbredde. ABR tjenesten inneholder en godt definert flytkontroll protokoll som tar sikt på å minimalisere pakketapet i nettverket. GFR og UBR tjenestene kontrollerer ikke flyten av trafikk, men er avhengig av at høyere lags protokoller (f.eks. Transmission Control Protocol – TCP) tar seg av flyt kontrollen.

6.1.1.1 Constant bit rate (CBR) tjenesten

Real-time applikasjoner inneholder vanligvis video eller lyd informasjon. Disse applikasjonene har strenge krav til forsinkelse, og mottaker kan være følsom for variasjoner i ende-til-ende transmisjons forsinkelsen. Trafikken fra applikasjonen kan være ujevn av natur, men peak båndbredden blir statistisk allokert for varigheten av forbindelsen selv om behovet for denne båndbredden ikke er konstant.

CBR tjenesten er designet for å støtte real-time applikasjoner. Den gir forbindelsen dedikert båndbredde og ekstrem lav sannsynlighet for pakke tap, samtidig som den gir liten og forutsigbar forsinkelse. Tiden mellom to celler som ankommer er konstant og kan bli karakterisert som minimal, som samsvarer med en kjent peak emisjons rate, kalt *peak cell rate* (PCR). CBR forbindelsen må ikke bruke den allokerte båndbredden hele tiden, men når det trengs blir den gjort tilgjengelig med en gang for å kunne møte kravene til forsinkelse og jitter. Generelt blir peak båndbredden til forbindelsen allokert statistisk og ingen statistiske fordeler blir oppnådd med denne tjenesten. Imidlertid kan en oppnå noen statistiske fordeler ved å overbooke ressursene [20].

6.1.1.2 Variable bit rate (VBR) tjenesten

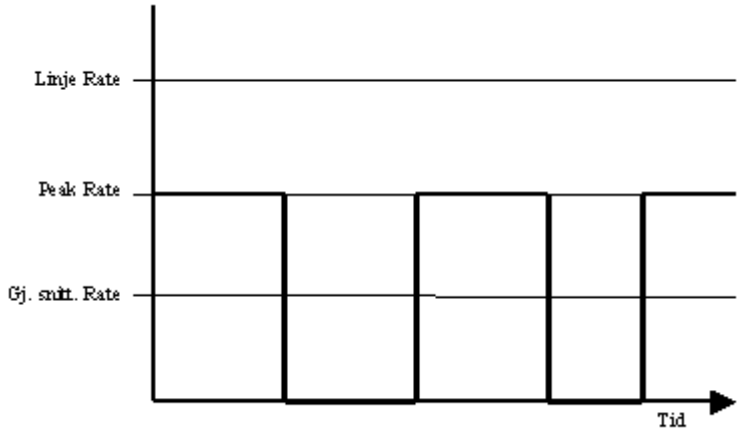
Tjeneste kategorien VBR er hovedsakelig tilsiktet for å gi en mer effektiv støtte av video applikasjoner og frame relay trafikk, eller andre applikasjoner som har kjent eller forutsigbar ujevn trafikk karakteristikk [26]. VBR trafikk kan bli karakterisert av en *sustained cell rate* (SCL) og en *peak cell rate* (PCR). SCR blir målt over en definert periode og representerer den gjennomsnittlige transmisjonsraten. PCR beskriver den minimale avstanden mellom celler, som representerer den nødvendige peak båndbredden. Generelt, tillater den gjennomsnittlige transmisjonsraten realiseringen av en statistisk fordel (jf. kapittel 6.2.1.), ved å statistisk allokere båndbredde lavere enn peak båndbredde og lik eller høyere enn den gjennomsnittlige båndbredden. VBR tjenesten er videre delt inn i to underkategorier basert på kravene til forsinkelse til applikasjonen [20]:

- Real-time VBR (rt-VBR)

- Non real-time VBR (nrt-VBR).

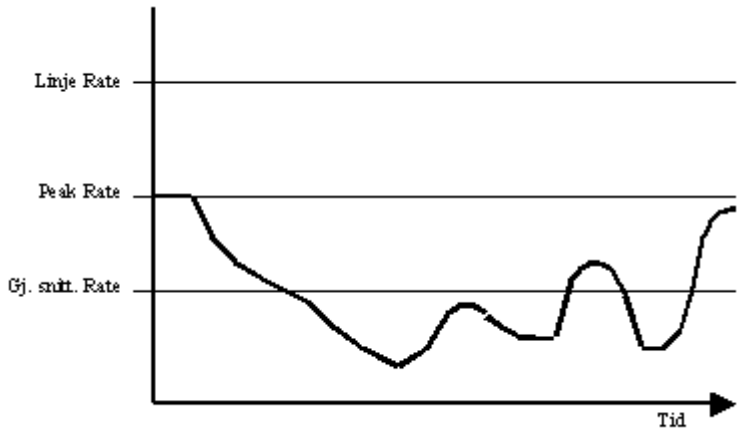
rt-VBR trafikken har strenge krav til ende-til-ende forsinkelse og kan derfor ikke bli buffret over lengre tid i nettverket. nrt-VBR trafikken garanterer ingen grenser på forsinkelsen og kan derfor buffres lenger i nettet. Differansen i buffer størrelsen har innflytelse på den båndbredden som blir allokert.

En data applikasjon som benytter seg av en nrt-VBR tjeneste, kan bli modellert som en PÅ/AV kilde [26]. Under PÅ perioden, kalt *burst*, blir celler generert med en konstant rate basert på en peak emisjonsrate. Ingen celler blir generert under AV perioden. Gjennomsnitts- og peak emisjonsrate kravene kan derfor bli karakterisert. PÅ perioden kan representere transmisjonen av en høyere lags protokoll data enhet (PDU) [26]. Figur 13 gir et eksempel på en PÅ/AV transmisjonsrate fra kilden, over tid. I dette eksemplet sender kilden frames med en peak emisjonsrate som er lavere enn den fysiske raten til mediet. Kilden er da stille i en periode, før den sender neste frame.



Figur 13: Eksempel på PÅ/AV.

Video er en typisk applikasjon for rt-VBR tjenesten. Den demonstrerer en variabel transmisjons rate uten PÅ/AV egenskapene til data applikasjonen. I dette tilfellet svinger raten fra kilden dynamisk, uten å overskride en kjent peak rate og en forutbestemt gjennomsnittlig emisjons rate. I dette eksemplet, kan en økning i transmisjonsraten bli forårsaket av en økning i bevegelser i video scenen. Figur 14 illustrerer denne oppførselen.



Figur 14: Eksempel på dynamisk rate.

Siden den gjennomsnittlige emisjonsraten er kjent med VBR tjenesten, kan statistiske fordeler bli oppnådd ved å allokere ressurser som er lavere enn peak emisjons raten og ved bruk av buffere for å håndtere potensielle overlapp av burster. CAC prøver å minimalisere mengde med båndbredde som blir allokert, mens den prøver å møte den angitte QoS.

6.1.1.3 Bandwidth-on-Demand tjenester

Noen data applikasjoner har ingen tidligere kjennskap til den påkrevde trafikk karakteristikken, før den setter opp forbindelsen. De har ingen krav til real-time forsinkelse, og er ikke fullt så følsomme for tap. Slike applikasjoner er gode kandidater til å benytte en av BoD tjenestene (ABR, GFR eller UBR). Ende-til-ende protokoller, for eksempel TCP, regulerer trafikkflyten basert på pakketap over ATM laget. Applikasjoner som benytter slike protokoller kan med fordel benytte BoD tjenester.

Available bit rate (ABR) tjenesten

ABR tjenesten kan garantere en minimum mengde med båndbredde og den kan bli begrenset til en spesifisert peak emisjons rate. ABR trafikk kilden deltar aktivt i en godt definert flyt kontroll protokoll, for å minimalisere sannsynligheten for celle- og rammetap i nettverket. Ved å bruke ABR tjenesten, bli metningen forskjøvet fra nettverket til dets ender. Båndbredde justeringen blir gjort via en spesifisert rate-basert flyt kontroll mekanisme, og alle forbindelser må følge protokollen for å kunne redusere celle tapet [26].

Garanteed frame rate (GFR) tjenesten

GFR tjenesten trenger ikke å henge fast på en flyt kontroll protokoll. Tjenesten garanterer en minimum båndbredde, men gjør ingen forpliktelser med hensyn på mengden av tap når applikasjonen overskrider den garanterte båndbredden. GFR tjenesten er designet for håndtere PDU'er fra lag over AAL (f.eks. AAL5). Nettverket sikter inn på å forkaste hele PDU'er i steden for å droppe celler tilfeldig ved metning. GFR tjenesten er ikke en del av [27] og er fortsatt under utvikling av ATM Forum og ITU-T [26].

Unspecified bit rate (UBR) tjenesten

UBR tjenesten er veldig enkel når det gjelder trafikk behandlings funksjonalitet. Dette betyr at UBR forbindelser deler den gjenværende båndbredden uten å bruke noen spesielle mekanismer for tilbakemelding. Med denne tjenesten, får applikasjonen så mye båndbredde som nettverket kan gi, men den må tolerere et uspesifisert celle tap. Denne tjenesten kalles ofte for en "best effort" tjeneste.

UBR tjenesten kan bli forsterket med mer sofistikerte trafikk behandlings former, som for eksempel *frame discard*, for å øke nettverkets evne til å bære hele pakker med data eller *goodput* [26].

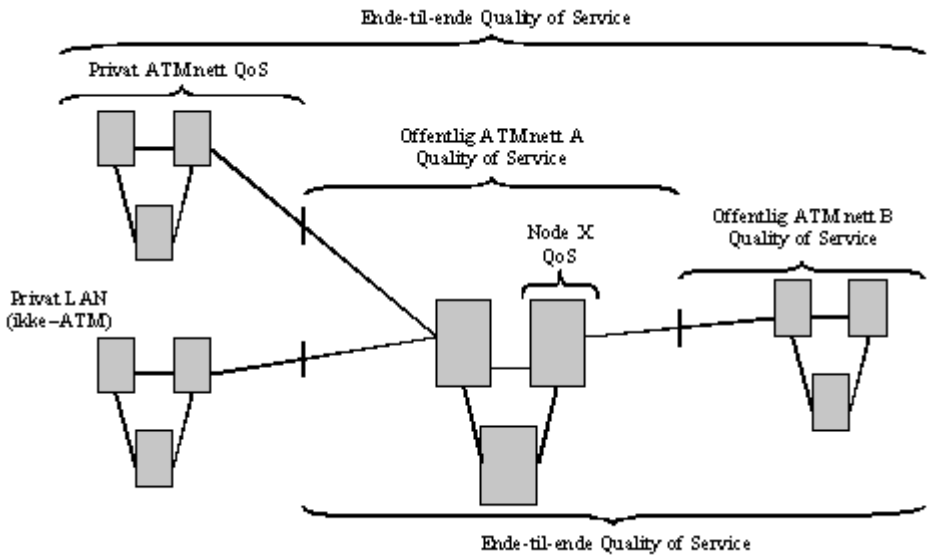
Den effektive QoS til UBR tjenesten kan bli administrert via kjente regler, for eksempel ved å begrense antall forbindelser som deler den resterende båndbredden.

6.1.2 QoS parametre

Den tjenestekvaliteten som finnes på ATM laget blir definert av et sett med parametre, som karakteriserer kravene til ytelsen til forbindelsen. Disse QoS parametrene bestemmer kravene til ytelsen ende-til-ende på ATM laget. Ende-til-ende ytelse, kombinerer ytelsen til alle elementene i nettverket. Figur 15 beskriver en ende-til-ende referansemodell, som viser hvordan QoS påføres node-, nettverks- og ende-til-ende nivået. Trafikk kontrakten definerer QoS ende-til-ende, d.v.s. mellom grensene til ATM nettet, men ekskludert ende systemene [26].

Seks QoS parametre blir brukt til å måle ytelsen til nettverket for en gitt forbindelse. Tre av disse kan bli forhandlet fram mellom ende systemene og nettverkene som en del av trafikk kontrakten [20]:

- *Cell loss ratio* (CLR)
- *Maxsimum cell transfer delay* (Max-CTD)
- *Peak-to-peak cell delay variation* (PTP-CDV)



Figur 15: En referanse modell.

Tre andre QoS parametre er ikke forhandlingsbare som en del av trafikk kontrakten [20]:

- *Cell error ratio* (CER)
- *Severely errored cell block ratio* (SECBR)
- *Cell misinsertion rate* (CMR)

6.1.2.1 Cell loss ratio (CLR)

Tap av celler oppstår fordi bufferne i nettverket blir fulle, som en følge av skurer med trafikk fra flere forbindelser. CAC planlegging og køstrategier, kan ha en effekt på tap av celler.

Tap av celler kan også oppstå når komponenter i nettet bryter sammen og når det blir svitsjet over til en redundant link. CLR blir definert for hver forbindelse som [26]:

$$CLR = \frac{\text{Tapte celler}}{\text{Totalt sendte celler}}$$

Likning 2

hvor det med tapte pakker menes:

- det antall celler, som ikke kom fram til mottaker
- det antall celler, som ble mottatt med feil header
- det antall celler, der innholdet var ødelagt av feil

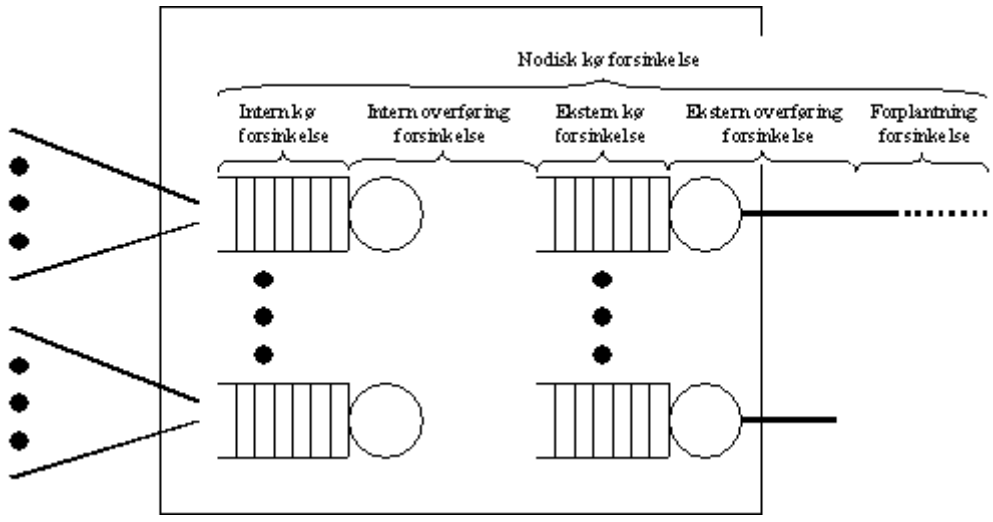
Totalt sendte celler, er det antall tilpassede celler som blir transmittert over en periode. CLR gjelder ikke for celler som ikke er tilpasset til trafikk beskrivelsene.

Måleperioden over er ikke standardisert, men generelt blir den representert ved livstiden til forbindelsen. For en permanent virtuell forbindelse blir perioden det måles over definert av nettverksoperatøren, og er vanligvis stor nok til gjelde for korte transiente perioder med metning.

CLR kan enten bli målt for cellene med CLP = 0 eller summen av cellene CLP = 0 og CLP = 1, som bestemt av tilpasningsdefinisjonen [26].

6.1.2.2 Forsinkelses parametere

Den målte *cell transfer delay* (CTD), er definert som tiden mellom en celle blir sendt fra det genererende ende systemet, til den kommer fram til destinasjonen [26]. Figur 16 viser komponentene til CTD i en node, med inn- og utkøer.



Figur 16: Komponenter i CTD.

CTD gjennom nettverket, er en sum av alle *cell transfer delays* ved hver node i banen. Summen av CTD ved hver node, omfatter i*nterne forsinkelser* (som en følge av en eller flere interne køpunkter, svitsjing, prosessering og intern transmisjons link),*ekstern køing* og *overførings forsinkelse* som oppstår før pakken forlater noden. Andre *prosesserings forsinkelser* kan bli lagt til disse. Når pakken er på linken, blir også *forplantnings forsinkelsen* lagt til CTD [26].

- Den *interne kø og overførings forsinkelsen*, er den tiden som trengs for å køe og sende bitene på den interne linken i noden. Fordelingen av køforsinkelsene varierer som en funksjon av belastningen og hvilken type planleggingsalgoritme (scheduling) som blir brukt. Svitsjens arkitektur kan kreve mange eller ingen interne kø punkter, med potensielt forskjellige interne link rater.

- Den *eksterne kø og overførings forsinkelsen* er den tiden som trengs for å køe og sende biter på det eksterne grensesnittet (eller ut linken). Fordelingen av kø forsinkelsen varierer som en funksjon av belastningen og hvilken type scheduling algoritme som blir brukt ved kø punktet.

- *Forplantnings forsinkelsen* er den tiden som kreves for at bitene skal forplante seg langs det fysiske mediet. En faktor av stor betydning for forplantnings forsinkelsen er tiden det tar for signalet å komme fra en node til en annen. Elektromagnetiske signaler forplanter seg med en hastighet på 0.2 til 0.6 av lysets hastighet, avhengig av hvilket medie som blir brukt. Denne faktoren kan være signifikant når det gjelder store avstander. Forplantnings forsinkelsen (FF) kan beregnes som [26]:

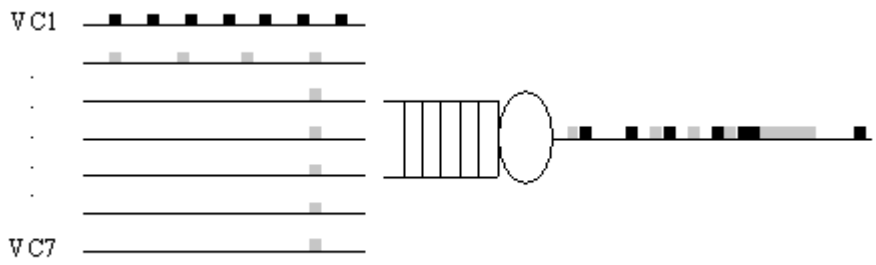
$$FF = \frac{\text{Avstand}}{\text{Forpl. faktor} \times \text{lysets hastighet}}$$

Likning 3

- *Prosesserings forsinkelsen* representerer forsinkelser som er nødvendige for å behandle hver celle (f.eks. å analysere headeren).

Minimum CTD blir satt sammen av de ikke variable elementene i CTD, det vil si den CTD som oppstår når en pakke går gjennom nettverket, uten å bli lagt i noen køer. Minimum CTD kan bli kalkulert som summen av den interne, eksterne, prosessering og forplantnings forsinkelsen over alle linkene i en node.

På grunn av den statistiske naturen til ATM, avhenger køforsinkelsen fra en celle til en annen, og på den måten oppstår det variasjon i målingen av forsinkelsen. Denne variasjonen blir kalt *cell delay variation* (CDV). På grunn av CDV, kan en konstant og jevn strøm av celler, inn til et køpunkt, resultere i en ujevn strøm av celler. Dette fenomenet påvirker for eksempel størrelsen på bufferen i video mottakere og kan få innvirkning på kvaliteten (bilde og lyd) ved å utarme eller overbelaste mottakeren. Forskjellige svitsje arkitekturer og planleggingsalgoritmer har innflytelse på variasjonen i kø forsinkelsen.



Figur 17: CDV Illustrasjon.

Figur 17 viser hvilken innflytelse køing har på trafikk karakteristikken.

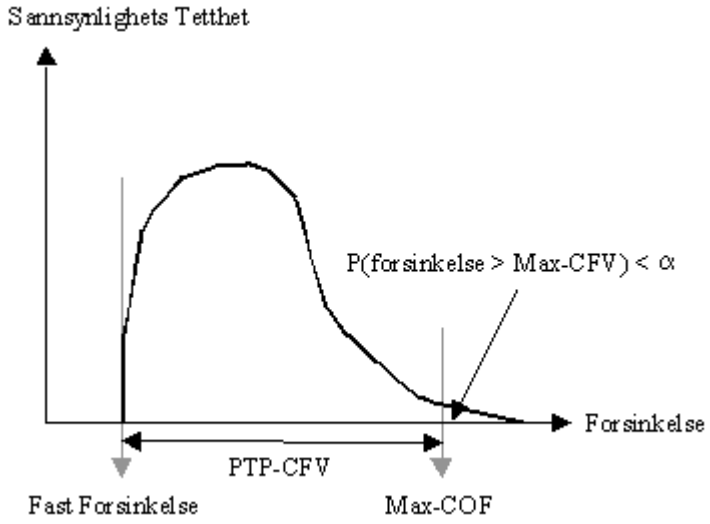
I dette eksempelet blir VC1 multiplekset med seks andre VC'er, med forskjellig stadfestet PCR. I dette eksempelet har inn- og utgangene samme hastighet. Ankomsten til de andre seks VC'ene er synkronisert, noe som gjør at det oppstår kø og forstyrrer dermed flyten til VC1.

CDV er et fenomen som oppstår i alle køer, delvis som en funksjon av bufferstørrelsen, men hovedsakelig som en følge av hvilken planleggingsstrategi (scheduling) som blir brukt [26]. QoS parameteren CDV, blir ofte forvekslet med *cell delay variation tolerance* (CDVT), som beskriver trafikk forbindelsen. CDVT er den maksimalt tolererte CDV for en gitt forbindelse, hvor trafikken ikke er tilpasset.

For å fange denne potensielle svekkelsen, når det gjelder oppnåelige QoS formål, har det blitt definert to ende-til-ende QoS parametere [26]:

- *Maximum cell transfer delay* (Max-CTD)
- *Peak-to-peak cell delay variation* (PTP-CDV)

Figur 18 gir et eksempel på sannsynlighets tetthetsfunksjonen til CTD og relateres til de nevnte parameterene. Sannsynlighets tetthetsfunksjonen i figur 18 henvender seg til peak raten til real-time trafikken, og kan ikke være representativ for forsinkelse som oppstår ved ikke real-time trafikk.



Figur 18: Sannsylighets tetthetsfunksjonen til CTD.

Maximum cell transfer delay (Max-CTD)

Maximum cell transfer delay (Max-CTD) representers av den (1 - α) kvantile av sannsynlighets tetthetsfunksjonen til CTD, hvor de cellene med forsinkelse, som overskrider dette maksimum, blir antatt å være tapt eller ubrukelige. Max-COF inneholder de faste forsinkelses komponentene.

Tilordningen av α parameteren er nettverksspesifikk, men kan blir gjort ved følgende prosedyre. Anta at $maxCTD_q$ representerer forsinkelsesbudsjettet allokert til en kø i banen. For å kunne holde dette budsjettet, kan køen bli begrenset til en buffer størrelse B, hvor

$$B = \frac{\text{maxCTD}_q}{\text{Kø Tjeneste Rate}}$$

Likning 4

og hvor *kø tjeneste raten* representerer raten som køen blir tjent med. CAC funksjonen bruker denne kø størrelsen til å allokere nok båndbredde, for å sikre at sannsynligheten, for å overskride denne køen, ikke overgår CLR. Sannsynligheten for å overgå $maxCTD_q$ kan ikke overskride CLR, siden celler blir droppet når størrelsen på køen bli større en B. Derfor er α satt til CLR; en konservativ tilordning. Det er sannsynlig at køen ikke blir belastet over en periode, slik at CLR blir nådd og Max-CTD kan oppnås med en lavere kvantil enn CLR eller en lavere Max-CTD kan oppnås ved α = CLR [26].

Peak-to-peak cell delay variation (PTP-CDV)

PTP-CDV representerer differansen mellom maksimum og minimum CTD (d.v.s. Max-CTD minus de faste forsinkelses komponentene). Denne målingen gjør at en kan evaluere den maksimalt mulige forsinkelsen mellom to etterfølgende celler, som ble deterministisk skilt. En kan også estimere den verst mulige forekomsten av skurer, som en følge av køing.

Severely errored cell block ratio (SECBR)

Severely errored cell block ratio (SECBR) er definert for en forbindelse som følger:

$$SECBR = \frac{\text{Severely errored cell blocks}}{\text{Total transmitted cell blocks}}$$

Likning 5

En celle blokk er en sekvens av N celler, som overføres etter hverandre på en gitt forbindelse. En hard skadd celle blokk oppstår når mer enn M celler med feil, tapte celler eller feilinnførte celler, blir observert i en mottatt celle blokk. Av praktiske grunner tilknyttet målinger, tilsvarer en celle blokk normalt det antall av celler med bruker informasjon som blir sendt mellom OAM (Operation Administration og Maintenance) celler [26]. Tapte og sendte celler som fins i en hardt skadd celle blokk blir ekskludert fra beregningen av CLR.

Cell misinsertion rate (CMR)

CMR er definert for en forbindelse som følger:

$$\text{CMR} = \frac{\text{Feilinnfør te celler}}{\text{Tids intervall}}$$

Likning 6

En feilinnført celle er en celle som blir overført over en VC, som den ikke hører hjemme på. Feilinnføring av celler på en bestemt forbindelse skyldes ofte en uoppdaget feil i headeren, som fører til at cellen blir overført på feil forbindelse. Denne ytelsesparameteren er definert som en

rate (og ikke et forhold), siden mekanismen som skaper feilinnførte celler er uavhengig av det antall overførte celler som er mottatt på den tilsvarende forbindelsen.

Feilinnføringsraten påvirkes først å fremst av feil som ikke blir oppdaget eller rettet i headeren til cellen, som igjen er påvirket av feilraten til overføringen. Sannsynligheten for at en feil som ikke er oppdaget eller rettet i headeren til cellen kommer inn på en VPI/VCI, er også avhengig av det antall VPI/VCI verdier som er tildelt og blir aktivt brukt. De hard skadde celle blokkene blir ikke tatt med i beregningen av *cell misinsertion rate*.

Cell error ratio (CER)

CER er definert for en forbindelse som følger:

$$\text{CER} = \frac{\text{Celler med feil}}{\text{Celler som er overført uten feil} + \text{Celler med feil}}$$

Likning 7

En celle med feil, er en celle som har fått innholdet (header eller payload) modifisert feilaktig, slik at innholdet ikke kan fåes tilbake ved hjelp av feilkorreksjons teknikker. CER påvirkes av feilkarakteristikken til det fysiske mediet og distansen til mottaker.

6.1.3 Trafikk deskriptorer

En ATM forbindelse karakteriserer sin trafikk ved hjelp av trafikk deskriptorer fra kilden. Deskriptorene forsøker å fange ankomstmønsteret til cellene, for å benytte det til allokering av ressurser (jf. kapittel 6.2). Trafikk deskriptorene for en ATM forbindelse inneholder en eller flere av følgende [26]:

- *Peak cell rate* (PCR)
- *Sustainable cell rate* (SCR) og *maximum burst size* (MBS)
- *Minimum cell rate* (MCR)
- *Maximum frame size* (MFS)

Hver forbindelse har to sett med trafikk deskriptorer, som beskriver toveis trafikk karakteristikk. Trafikk deskriptorene, som blir gitt ved oppkoblingen av forbindelsen, varierer avhengig av tjenestekategorien til forbindelsen. Alle trafikk deskriptorene fra kilden, kan bare bli spesifisert for celler med CLP = 0, eller for de samlede cellene i forbindelsen, uten hensyn til CLP biten (CLP = 0 + 1).

Under oppkoblingen av en forbindelse, blir det lagt til en *cell delay variation tolerance* (CDVT) parameter til trafikk deskriptorene fra kilden. Trafikk deskriptorene til forbindelsen inneholder deskriptorene fra kilden og CDVT. CDVT blir brukt for å verifisere tilpasningen av trafikk deskriptorene fra kilden.

6.1.3.1 Peak cell rate (PCR)

Peak cell rate (PCR) representerer peak emisjonsraten til kilden. Den inverse til PCR gir det teoretiske minimum av ankomst tiden til cellene, for en gitt forbindelse. PCR kan bli begrenset

av den fysiske hastigheten (eller link raten) til kilden eller gjennom forming av inngangs-trafikken. PCR blir uttrykt i celler pr. sekund. Figur 19 beskriver trafikk karakteristikken for en forbindelse som har fått en PCR på en tredjedel av linje raten. Hver tidsluke tilsvarer den tiden det tar å sende en celle ved den linje raten. En forbindelse med PCR på en tredjedel av linje raten, kan sende en celle hver tredje tidsluke.

Med unntak av ABR, blir PCR alltid definert som den aggregate (CLP = 0 og CLP = 1) celle flyten [20]. Det vil si at cellene i figur 19 er enten CLP = 0 eller CLP = 1. En forbindelse kan ikke

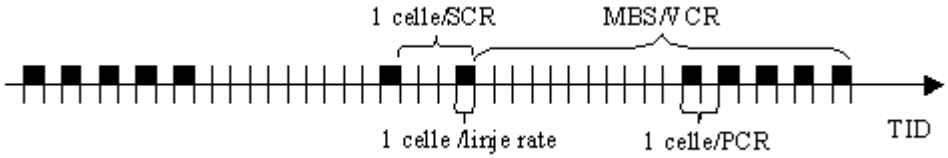
Figur 19: Peak cell rate.

overskride PCR ved å sende celler med lavere prioritet (CLP = 1). I tilfellet med ABR, blir PCR bare definert på en celle flyt med CBR = 1.

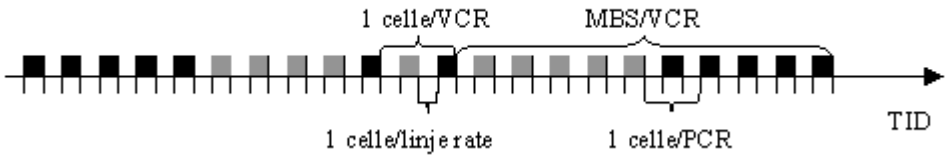
6.1.3.2 Sustained cell rate (SCR)

Sustained cell raten (SCR) er en øvre grense for den gjennomsnittlige overføringsraten til de tilpassede cellene, i en ATM forbindelse, som over en tidsskala er lang relativt til de som PCR er definert for [26].

Forbindelser som bruker VBR tjenesten kan definere en SCR. Den inverse av VCR, represen-terer den øvre grensen på den teoretiske gjennomsnittlige (langsiktig) ankomst tiden til cellene, med hensyn på link hastigheten. SCR blir alltid spesifisert med en tilsvarende *maximum burst size* (MBS). MBS parameteren representerer *burst faktoren* til forbindelsen. Den spesifiserer det maksimalt antall celler som kan sendes fra kilden med PCR, mens den fremdeles overholder den forhandlede SCR [26].



Figur 20: Sustained cell rate og maximum burst size.



Figur 21: Sustained cell rate og maximun burst size.

Figur 20 gir et eksempel på en forbindelse, som har fått en PCR på halvparten av linje raten og en SCR på en fjerdedel av linje raten, med en MBS på fem celler. Forbindelsen kan sende fem celler over ti tidsluker, men den forblir stille i ti tidsluker, for å overholde gjennomsnittsraten på

en fjerdedel av linje raten. Alternativt kan forbindelsen sende en celle hver fjerde tidsluke, men i dette tilfellet, vil dette ikke være å ta fordel av burst muligheten.

SCR kan bli definert for enten den aggregate (CLP = 0 og CLP = 1) celleflyten eller for bare CLP = 0 [26]. Når den blir definert for den aggregate av alle cellene, kan cellene i figur 20 være enten CLP = 0 eller CLP = 1. I dette tilfellet, kan en forbindelse ikke overskride SCR ved å sende lavere prioritets (CLP = 1) celler. Når SCR er definert på CLP = 0 flyten, kan forbindelsen overskride SCR ved å sende lavere prioritets (CLP = 1) celler opp til PCR. Figur 21 demonstrerer denne muligheten.

6.1.3.3 Minimum cell rate

Minimum cell rate (MCR) er klassifisert som en trafikk deskriptor, selv om den representerer minimum allokert båndbredde for en forbindelse. Den beskriver bokstavelig talt ikke oppførselen til trafikken, det vil si at forbindelsen kan sende trafikk med en mye høyere rate enn MCR. Den blir brukt for BoD tjenestene (ABR og GFR), for å sikre at forbindelsen ikke utarmes, når det ikke er noe mer tilgjengelig båndbredde. Siden den også representerer en garantert minimum throughput, kan MCR også bli betraktet som en QoS parameter [20].

6.1.3.4 Maximum frame size

Maximum frame size (MFS) definerer den maksimale størrelsen på en AAL PDU, som kan bli sendt på en GFR forbindelse. AAL frames, som overskrider denne størrelsen, er ikke kvalifisert til å motta GFR sin QoS [20].

6.1.4 Tjeneste klasser

For å kunne begrense mengden med kombinasjoner av QoS parametere og trafikk deskriptorer som støttes av ATM nettverket, har det blitt standardisert noen forhåndsdefinerte kombinasjoner for hver tjeneste kategori [20]. Tabell 1 beskriver disse kombinasjonene.

Attributt	CBR	Real-time VBR	Ikke real-time VBR	ABR	GFR	UBR
PCR	Spesifisert	Spesifisert	Spesifisert	Spesifisert	Spesifisert	Spesifisert
SCR, MBS	I/E	Spesifisert	Spesifisert	I/E	I/E	I/E
MCR	I/E	I/E	I/E	Valgfri	I/E	I/E
MCR, MBS, MFS	I/E	I/E	I/E	I/E	Spesifisert	I/E
CLR	Forpliktet	Forpliktet	Forpliktet	Ikke mål	Ikke mål	Ikke mål
Max-CRD	Forpliktet	Forpliktet	Ikke mål	Ikke mål	Ikke mål	Ikke mål
PTP-CDV	Forpliktet	Forpliktet	Ikke mål	Ikke mål	Ikke mål	Ikke mål

Tabell 1: QoS og trafikk deskriptorer per tjeneste kategori.

En *spesifisert* parameter blir gitt til nettverket ved oppkobling av forbindelsen. *Ikke egnet* (I/E) betyr at parameteren ikke er en del av trafikk kontrakten for denne tjeneste kategorien. En

valgfri parameter kan inkluderes som en del av trafikk kontrakten. Standard verdier kan bli brukt hvis parameteren ikke er inkludert. En *forpliktet* QoS parameter betyr at målet blir gitt ved oppkoblingen av forbindelsen og at nettverket utøver dette målet hvis forbindelsen bli akseptert. Hvis QoS parameteren *ikke har noe mål*, vil det si at parameteren ikke blir tatt med i betraktiningen, av beslutningen om å akseptere forbindelsen. Nettverket gjør ingen kvantitative/målbare forpliktelser vedrørende denne parameteren og forplikter seg bare til å levere cellen så fort som mulig (CTD og CDV) eller hvis det er mulig (CLR) [26].
For en gitt tjeneste kategori, kan et nettverk tilby en eller flere tjeneste klasser (CoS) [7]. En tjenesteklasse tilbyr er sett med QoS mål og kan begrense rekkevidden til noen av trafikk deskriptorene. For eksempel kan et nettverk tilby to QoS klasser med CBR tjeneste, en meget bra tjeneste med et CLR på 10^{-10} og en normal tjeneste med CLR på 10^{-7}. Den nedre kvalitetsklassen kan kreve mindre allokering av båndbredde. Forskjellige tariffstrukturer kan gis til forskjellige tjenesteklasser, og på denne måten gi en høy fleksibilitet i måten å designe kost effektive nettverk. Det er ingen betingelser på hvilket sett med tjenester eller tjenesteklasser som støttes i nettverket [26].

6.1.5 Å forhandle fram trafikk kontrakten med nettverket

Dette kapittelet beskriver hvordan trafikk kontrakten blir forhandlet fram med nettverket. Når en setter opp *permanent virtual connections* (PVC) eller *permanent virtual paths* (PVP), blir innstillingen av de forskjellige elementene i trafikk kontrakten gjort via nettverks administrasjons systemet. For svitsjede VC'er eller VP'er, er det signalerings protokollen som tar seg av forhandlingen av forbindelses parameterene [26].

PCR og SCR parameterene blir signalert i celler pr. sekund. MBS blir signalert i celler. Kodens oppløsning har innflytelse på den mengden med båndbredde som blir allokert og fleksibiliteten med fastsettelse av forskjellige rater.

Hvis CAC til en node i banen ikke kan allokere den båndbredden som trengs, for å kunne støtte de trafikk deskriptorene som er bedt om, kan noden redusere verdien til trafikk deskriptoren [20]. Imidlertid kan den ikke redusere verdien til trafikk deskriptorene under det som blir spesifisert som minimum akseptabel verdi i signalerings meldingen. Hvis verdien som kan håndteres av nettverket faller under dette minimum, blir forbindelsen avvist.

PTP-CDV og Max-CTD blir signalert med enheter som er oppgitt i microsekunder. CLR blir uttrykt som en størrelse på *n* ($1 < n < 15$), hvor CLR målet får verdien 10^{-n}.

Når signalerings meldingen forplanter seg fra node til node, blir Max-CTD og PTP-CDV samlet og overført i signalerings meldingen, ved siden av den begjærte verdien. Når signalerings meldingen kommer til den siste noden, representerer de lagrede QoS parameterene det ende-til-ende betingelsene som kan gis av nettverket. Oppkoblingen av forbindelsen bil avvist hvis de akkumulerte verdiene overskrider de begjærte betingelsene [20]. Metoden for å akkumulere Max-CTD og PTP-CDV gjennom nettverket har en direkte innvirkning på effektiviteten til nettverket og blir diskutert under.

CLR parameteren blir ikke tydelig akkumulert. Hvert element i nettverket aksepterer eller avviser forbindelsen basert på en sammenlikning av tapsraten i nettverkselementet og den begjærte CLR. CLR blir også vanligvis brukt som en øvre grense på parameteren, som blir brukt

til å beregne forsinkelses betingelsene. Et nettverk kan velge en mindre α (større sannsynlighet), som kan ha den effekten at en overestimerer den kumulative Max-CTD og PTP-CDV.

Additiv akkumulasjon av PTP-CDV og Max-CTD

En enkel akkumulasjon, som ikke tar hensyn til de potensielle køpunktene, kan føre til en undervurdering av ende-til-ende forsinkelsen og nettverket kan akseptere forbindelser der den begjærte QoS ikke kan møtes [26].

En ”worst-case” tilnærming av PTP-CFV er som følger. En svitsj mottar den akkumulerte PTP-CDV og legger ”worst-case” CDV til PTP-CDV.

Likevel er ikke PTP-CDV additiv, det vil si at den økes eller reduseres fra node til node. Ved å summere opp ”worst-case” peak-til-peak målingene får en en tryggere øvre grense, men kan få den effekten at en avviser mange forbindelser når nettverket vokser i diameter. Mer sofistikerte akkumulasjons teknikker, gir et bedre estimat av ende-til-ende forsinkelse. Den kan derfor redusere det antall forbindelser som avvises, men de har en tendens til stole på antagelser om ankomstmønsteret til cellene eller trafikk belastningen som kan sette QoS på spill. Deres kompleksitet, kan også ha innflytelse på oppkoblingsraten [26].

En enkel akkumulasjon for Max-CTD er som følger. En svitsj mottar den akkumulerte Max-CTD og legger til sitt Max-CTD bidrag, til den akkumulerte Max-CTD. Denne additive metoden er også en ”worst-case” antagelse, siden sannsynligheten for at en celle lider Max-CTD ved hver svitsj, er svært liten.

6.2 Connection admission controll

En ATM forbindelse traverserer en rekke svitsjende noder i nettverket. Selv inn i en svitsjende node kan en forbindelse gå igjennom et antall køpunkter. For å sette opp en forbindelse på en slik bane, må det reserveres ressurser i hvert kø punkt, for at en skal kunne garantere den fastslåtte tjenestekvaliteten. Generelt så settes det av buffer plass og den båndbredden som trengs for at en skal kunne tjene en forbindelse i køpunktet. De sett med regler eller prosedyrer som bestemmer om en forbindelse skal få tilgang i en ATM svitsj kalles *connection admission controll* (CAC) [20].

Som diskutert i kapittel 6.1, kan en ATM forbindelse bære trafikk med en bestemt tjeneste kategori (f.eks. CBR, VBR, ABR, GFR og UBR). Videre så besitter hver av disse tjeneste kategoriene forskjellige QoS mål. Det er da rimelig at CAC reglene er forskjellig fra hver tjenestekategori.

For å verifisere om det kan gis adgang til en forbindelse, følger CAC, prosedyren som er beskrevet under for å sette opp en forbindelse ved hvert kø punkt [26,31]:

1. Kartlegge trafikk deskriptorene, som er assosiert med en forbindelse, inn i en trafikk modell. Siden hver tjeneste kategori assosieres med spesifikke trafikk deskriptorer, er det nødvendig med forskjellige trafikk modeller.

2. Bruk denne trafikk modellen med en passende kø modell for å estimere om det er nok ressurser i systemet for å gi forbindelsen adgang og den begjærte QoS.

3. Allokere ressurser hvis de kan overholde kravene og gi adgang til forbindelsen.

Avhengig av hvilken trafikk modell som blir brukt, kan CAC prosedyrene være for konservative ved å allokere for mye ressurser. Dette reduserer de *statistiske fordelene* (som er definert i kapittel 6.2.1) som kan oppnåes. En effektiv CAC lager maksimum statistisk fordel, uten å bryte QoS. Effektiviteten til CAC er derfor avhengig av hvor nært virkeligheten trafikk og kø modellene kommer [26].

Det bør nevnes at CAC algoritmene ikke kan være beregningsintensive, fordi de blir utført i real-time av ATM svitsjene [26]. Denne utførelsen påvirker oppkoblingsraten og oppkoblingsforsinkelsen til forbindelser direkte, fordi CAC algoritmen kjøres under hver oppkobling av en forbindelse. Siden forbindelser kobles opp og avsluttes i real-time, er det avgjørende å håndtere en høy oppkoblingsrate, for å kunne ha en effektiv støtte av svitsjede virtuelle kretser.

Det finnes andre teknikker for å allokere båndbredde basert på heuristikker eller målinger over lang tid. Denne typen CAC prosedyrer kan ikke garantere QoS og tillater ikke en sikker måte å allokere for mye ressurser på.

CAC algoritmene er ikke spesifisert av ATM Forum eller ITU-T, fordi hver svitsj arkitektur, kø- og schedulingimplementasjon kan være mer passende, for en spesiell type CAC implementasjon. Andre begrensninger, som prosesseringskapasitet eller buffer størrelse, kan kreve bruk av en bestemt CAC implementasjon. Det er ikke nødvendig å ha den samme CAC funksjonen i hver svitsj eller på hvert køpunkt inne i en svitsj, for å oppnå ende-til-ende QoS garantier.

6.2.1 Statistiske fordeler

Siden en forbindelses maksimale data rate, er *peak cell rate* (PCR), hvis en ser bort fra jitter (eller CDV), er en CAC nødt til å allokere mer enn PCR til en forbindelse. Siden en forbindelse ikke sender data kontinuerlig, er det mulig å allokere færre ressurser når mange forbindelser blir multiplekset ved et køpunkt. Dette betyr at det er mulig med statistiske fordeler og at flere forbindelser kan få adgang, enn hvis PCR bli allokert til hver forbindelse. Begrepet *statistisk fordel* kan bli definert som [26]:

$$\text{Statistisk fordel} = \frac{\text{Antall forbindeler som slippes inn med statistisk multipleksing}}{\text{Antall forbindelser som slippes inn med peak rate allokering}}$$

Likning 8

En effektiv CAC bør prøve å oppnå en så stor statistisk fordel som mulig uten å risikere å gå i mettet tilstand, som kan svekke QoS [26]. Fordelen, er generelt en funksjon av buffer størrelsen, trafikk karakteristikken og QoS målet til forbindelsene, som blir multiplekset. For et gitt sett med trafikk- og QoS-parametere, kan en oppnå større statistiske fordeler ved større buffer størrelser, opp til et visst punkt, hvor det begynner å gå tilbake igjen. Forekomsten av metning ved et kø punkt, kan bli analysert i to deler [26]:

1. *Celle-skala metning*, som oppstår i små buffere som en følge av at celler ankommer samtidig fra forskjelige forbindelser.

2. *Burst-skala metning*, som typisk oppstår i stor buffere som en følge av at burster med celler ankommer fra forskjellige forbindelser.

CBR og real-time VBR (rt-VBR) tjeneste kategoriene har node spesifikke forsinkelses krav. Det betyr at de node spesifikke forsinkelsene som erfares av cellene fra disse tjenestene, bør ikke være større enn en gitt verdi D (f.eks. 250μs) med en gitt kvantil Q (f.eks. 10^{-10}). Dette betyr, P(node spesifikk forsinkelse > D) ≤ Q. Dette forsinkelses kravet tvinger buffer størrelsen til å bli liten, og *celle-skala metning* vil være rådende for disse tjenestene. Det er derfor vanskelig å oppnå store statistiske mulitpleksende fordeler for CBR og rt-VBR tjenester. For nrt-VBR gir ATM svitsjene større buffere for å kunne absorbere burstene, og *burst-skala metning* vil forekomme ofte for disse tjenestene. Det er mulig å oppnå store statistiske fordeler for nrt-VBR tjenester.

6.2.2 CAC for CBR trafikk

En ren CBR trafikk kilde sender ut en celle periodisk hvert 1/PCR enhet. Hvis en ser bort fra CDV og celle-skala metning, er en enkel regel for CAC, å tildele PCR som båndbredde for hver CBR forbindelse og slippe inn forbindelser slik at [26]:

$$\sum_i PCR_i \leq \text{Link kapasitet}$$

Likning 9

Denne CAC algoritmen er basert på ”peak rate allokering”. Når ATM celler forplanter seg gjennom forskjellige køpunkter, vil periodisiteten til en ren CBR kilde gå tapt. Cellene kan klynge seg sammen eller spre seg, som en følge av buffring og påvirkning fra annen hindrende trafikk (f.eks. CDV).

Når celler klynger seg sammen, vil den effektive peak raten bli høyere enn kildens faktiske peak rate. Som en følge av CDV, vil ikke denne enkle CAC regelen være nok for å sikre at CLR til forbindelsene, som har fått adgang, er innenfor de fastsatte grensene. Selv uten denne jitteren, kan samtidig ankomst av celler, som en følge av multipleksing av periodiske celle strømmer, føre til celle tap. Fulle buffere, vil typisk skje i små buffere.

Generelt er det to tilnærminger som gjelder for CDV [26]:

- *Negligible CDV metoder*
- *Nonnegligible CDV metoder*

De *negligible CDV metodene* tar ikke direkte med CDV i betraktningen og antar at jitteren i trafikk strømmen er ubetydelig. Den *nonnegligible CDV metoden* modellerer multipleksingen av rene CBR, jevne (ikke jitter) kilder.

6.2.3 CAC for VBR trafikk

Som nevnt tidligere blir den *variable bit rate* (VBR) trafikken generelt karakterisert av en gjennomsnittlig rate (SCR), *peak celle rate* (PCR), *cell delay variation tolerance* (CDVT) og *maximum burst size* (MBS). QoS spesifikasjonene for rt-VBR trafikk er *cell loss ratio* (CLR) og forsinkelse, mens nrt-VBR trafikk har bare celle tapet [26].
Forholdet SCR/PCR definerer *burstiness* til en VBR forbindelse og har stor innflytelse på den statistiske fordelen. Hvis SCR/PCR << 1, vil en CAC basert på peak rate allokering være ueffektivt og konservativt. Det er vist at utnyttelsen av linken, kan gå så lavt som 5% når SCR/PCR er liten. Selv om bufferen for VBR trafikk er svært liten, er det unødvendig med peak rate allokasjon. Der er mulig å slippe inn forbindelser slik at:

$$\sum_i a_i \leq \text{Link kapasitet}$$

Likning 10

hvor, $SCF_i \leq \alpha_i \leq PCR_i$ for en forbindelse *i*. Da peak rate allokasjonen ikke gir noen statistisk fordel, gir ikke likningen over noen statistisk fordel. Den statistiske fordelen ved et kø punkt kan da bli definert som forholdet $\sum PCR_i / \sum \alpha_i$. α_i blir kalt den *effektive båndbredden, ekvivalente båndbredde* eller den *virtuelle båndbredden* til en forbindelse.

I tilfellet med liten eller ingen buffer, kan det bli brukt en *rate envelope multiplexsing* (REM) [26]. Dette betyr at forbindelser får adgang, slik at den totale aggregate ankomst raten (*rate envelope*) til en forbindelse, med høy sannsynlighet er mindre en link kapasiteten. På den andre siden, hvis det er en stor buffer ($\rightarrow \infty$) tilgjengelig for VBR trafikk, vil den absorbere bursten, slik at å allokere mer båndbredde enn SCR for hver VBR forbindelse, vil være unødvendig. I praksis vil buffer størrelsen være endelig og båndbredden vil falle mellom SCR og PCR. Denne teknikken kalles *rate sharing* (RS).

6.2.3.1 Rate envelope multiplexsing (REM)

Rate envelope multiplexsing (REM) metoden antar at det er liten eller ingen mulighet for buffring for VBR trafikk. Denne metoden blir derfor kalt *null buffer* eller den *bufferfrie* tilnærmelsen [26].

Denne metoden passer bra for real-time trafikk, på grunn av at det antas å være små buffere, og den modellerer celle-skala metning godt. Ved hjelp av REM metoden, får forbindelsen adgang, slik at den samlede ankomst raten til en forbindelse, med stor sannsynlighet, er mindre enn link kapasiteten.

6.2.3.2 Rate sharing (RS)

REM metoden er avhengig av den totale input raten ikke overskrider link kapasiteten eller at sannsynligheten for det er liten. Hvis denne antakelsen ikke er sann, må bufferene absorbere uoverensstemmelsen med raten. Dette gjelder spesielt for skurete trafikk, der gjennomsnittsraten er liten i forhold til peak raten.

For eksempel, hvis en har VBR trafikk med SCR << PCR, kan det være et mindre antall med forbindelser som sender med PCR, men summen av PCR kan være mye større enn link kapasiteten. For å garantere en viss QoS til forbindelsene, som f.eks. CLR, må en buffer lagre trafikken midlertidig og link kapasiteten er delt mellom forbindelsene. Derfor bør kømodellene bli lagt inn adgangsreglene til forbindelsen. Det er to fremgangsmåter å gjøre dette på:

1. Betrakte alle trafikk strømmer som multiplekses sammen og estimere sannsynligheten for celle tap, for å få den maksimale statistiske fordelen.

2. Betrakte hver forbindelse uavhengig og estimere båndbredde kravene til forbindelsen, som er slik at den kan garantere en gitt QoS. Denne tilnærmingen blir også kalt, *effektive båndbredde*.

Den *effektive båndbredden* bli omtalt i kapittel 6.2.3.3. Den første tilnærmingen betrakter forbindelsene som allerede har fått adgang, pluss den som er under vurdering sammen, og undersøker om den nye forbindelsen vil krenke QoS garantiene til noen av andre forbindelsene. Det vil si at denne metoden overveier tilstedeværelsen av andre forbindelser, for å fastsette QoS målene. Den nye forbindelsen blir akseptert hvis QoS målene blir holdt.

6.2.3.3 Effektiv båndbredde

Tilnærmingen som kalles *effektiv båndbredde* ser på hver forbindelse isolert, som om den var alene ved køpunktet [26]. Denne modellen kartlegger trafikkparameterene til hver forbindelse, til et reellt tall α_i, som kalles den *effektive båndbredden* eller den *ekvivalente båndbredden* til en forbindelse, slik at QoS kravene er tilfredsstilt. På den måten bli den *effektive båndbredden* utledet som en egenskap fra kilden, og med denne kartleggingen blir CAC regelen svært enkelt:

$$\sum_i a_i \leq \text{Link kapasitet}$$

Likning 11

Det vil si, at en forbindelse får adgang hvis det er tilgjengelig kapasitet, ellers blir den avvist. For en forbindelse med en gjennomsnitts rate SCR_i og en peak rate PCR_i, er den *effektive båndbredden* et tall mellom SCR_i og PCR_i. Det vil si $SCR_i \leq \alpha_i \leq PCR_i$. Verdien til den *effektive båndbredden* er avhengig av de statistiske egenskapene til forbindelsen, så vel som kø-egenskapene i metnings punktet.

Generelt for en gitt forbindelse sier det seg selv at den *effektive båndbredden* vil være nærme PCR, for små buffere og nærme VCR, for store buffere.

To egenskaper utgjør hovedfordelen ved denne metoden [26]:

1. *Additive egenskaper*. Effektive båndbredder er additive, det vil si at den totale effektive båndbredden som trengs for *N* forbindelser, er lik summen av de effektive båndbreddene til hver forbindelse.

2. *Uavhengighets egenskaper*. Den effektive båndbredden for en gitt forbindelse, er bare en funksjon av forbindelsens parametere.

Den additive egenskapen til denne metoden, gjør at den er meget akseptert og brukt i ATM teknologi. Siden forbindelser bli satt opp og revet ned dynamisk, med den effektiv båndbredde metoden, kan CAC funksjonen legge til (eller trekke fra) den effektive båndbredden til en forbindelse som blir oppkoblet (eller avsluttet), til den totale effektive båndbredden. Dette er ikke lett med metoder som ikke har uavhengighets- eller additivegenskapene. Imidlertid bør det nevnes at denne metoden kan være mer konservativ enn en metode som tar i betraktning den sanne statistiske multipleksingen (d.v.s. en metode som tar med andre forbindelser i betraktningen). Hovedgrunnen til dette er at den egentlige båndbredden som trengs for å kunne tjene *N* forbindelser, kan være mye mindre en summen av båndbreddene til *N* forbindelser.

6.2.4 CAC for ABR, UBR og GFR trafikk

CAC prosedyrene for ABR, GFR og UBR er ganske enkle. Både ABR og GFR har *minimum cell rate* (MCR) garantier. Derfor må CAC funksjonen tildele denne båndbredden for hver forbindelse. Basert på den nåværende definisjonen av GFR og avhengigheten av tilgjengelig buffer størrelse og håndteringsteknikker for buffere, må CAC ta med MFS og MBS parameterene i betraktningen [26].

Siden UBR trafikk ikke har båndbredde garantier, kan CAC teoretisk gi forbindelser adgang uten grense, men den kan designes slik at den begrenser antallet UBR forbindelser slik at hver forbindelse får en viss grad throughput i det lange løp.

6.2.5 Adgangskontroll for flerklasse trafikk

De CAC funksjonene som til nå er diskutert, er designet for en enkel tjenestekategori. Til vanlig består trafikkflyter av flere tjenesteklasser, der tjenestene kan være partisjonert og køet separat. Selv innen en gitt tjenestekategori, kan flere underklasser bli differensiert avhengig av deres QoS betingelser [26].

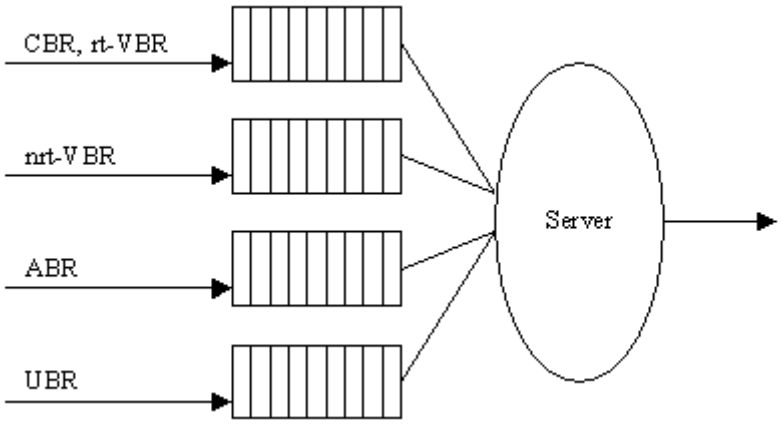
For å garantere en QoS for flerklasse trafikk, er det reservert en viss mengde båndbredde for hver tjenesteklasse. Med en effektiv båndbredde tilnærming, blir denne oppgaven svært enkel. La N_j være antall kilder for klasse *j*. La det være *K* slike klasser. CAC for flerklasse trafikk skal da sjekke at den totalt estimerte kapasiteten er mindre enn tjeneste raten, det vil si:

$$\sum_{j=1}^{K} N_j a_j \leq \text{Link kapasitet}$$

Likning 12

Prioritets køing, er en enkel måte å få en struktur, med forskjellig QoS. Et eksempel på dette er vist i figur 22. Likningen over er tilstrekkelig for å garantere QoS av en slik struktur, så lenge det er få eller ingen CLP = 1 celler fra CLP-signifikante VBR tjenester.

Berger og *Whitt* så på modifiseringer av effektiv båndbredde for prioriteringer. Generelt har trafikk med høy prioritet strengere ytelses kriterier enn trafikk med lavere prioritet. Det er derfor mulig å slippe inn flere forbindelser med trafikk av lavere prioritet. I steden for å benytte den lineære tvangen, som er brukt i likningen over, bruker metoden til Berger og Whitt et sett med



Figur 22: Eksempel på prioritetskøing.

kriterier. Hvis det er *N* nivåer av prioriteringer, trengs det *N* slike kriterier. Ved hvert nivå *k*, tar kriteriet i betraktning den effektive båndbredden til forbindelser ved høyere nivåer, som er sett ved nivå *k*. La α_{ij} betegne den effektive båndbredden til en prioritet *i*, klasse *j* forbindelse som ses av prioritet *k*, og n_{ij} være antall forbindelser med prioritet *i*, klasse *j*. Da er båndbredde kriteriene definert som:

$$\sum_{i=1}^{k}\sum_{j=1}^{J_i} a_{ij}^{k} n_{ij} \leq \text{Link kapasitet}$$

Likning 13

Her betegner *i* prioritets nivået og *j* tjeneste klassen, og $1 \leq j \leq J_i$. Ved å gjøre dette, blir det vist at det antall forbindelser som slippes inn er større enn det antall som slippes inn ved den lineær tvangen til likning 12.

6.2.6 CAC basert på målinger

De CAC prosedyrene som er beskrevet så langt er basert på analytisk modellering av oppførselen til både trafikk kilden og køstrukturene. I realiteten finnes det forskjellige trafikk kilder, som oppfører seg forskjellig, og det er umulig å modellere alle disse nøyaktig. Analytiske metoder, som blir brukt for å estimere ressursene som trengs for en gitt klasse med trafikk kilder, kan over- eller underestimere ressurskravene til en annen klasse.

Det er også mulig at kilder ikke utnytter trafikk deskriptorene. CAC prosedyrer som er basert på real-time målinger, kan da forutse ressursbruken mer nøyaktig. Disse prosedyrene måler visse ressurser i real-time og prøver å estimere om QoS kravene kan opprettholdes hvis en ny forbindelse får adgang.

6.2.7 Innstilling av CAC

Applikasjoner utleder de nødvendige trafikk deskriptorene basert på forsinkelse eller responstidsbehov, mens den forhandler kosten til forskjellige forbindelses båndbredder. Det er lite sannsynlig at forbindelsen trenger hele den forhandlede båndbredden på kontinuerlig basis. Data applikasjoner bruker båndbredden til å overføre store mengder data. Mindre mengder med data kan også utveksles periodisk, men bruken er generelt mindre enn SCR. Når det gjelder permanente forbindelser (PVC'er), kan den faktiske utnyttelsen sammenliknet med den forhandlede SCR være mange ganger mindre når den blir målt over levetiden til en forbindelse. For svitsjede forbindelser (SVC'er) er denne differansen vanligvis mindre, siden forbindelsen avsluttes når transaksjonen av informasjon er ferdig [26].

Utnyttelsen av den allokerte båndbredden, sammen med at CAC allokerer båndbredde konservativt, kan føre til mindre effektivitet med respekt på nettverkets evne til å bære CBR og VBR trafikk. Med passende målinger av ressurs utnyttelsen, er de mulig å bestemme graden av utnyttelse og stille inn CAC funksjonen riktig.

CAC sine *booking* og *skalerings faktorer* er generelt tilgjengelige, for å kunne tillate den faktiske bruken av båndbredden, å være med i betraktningen når en kunstig reduserer båndbredden, som statistisk allokeres til hver forbindelse. På den måten tillater en at flere forbindelser får adgang [26]. Det bør nevnes at den statistiske fordelen som oppnås gjennom bookning og skalerings faktorer, er helt forskjellig fra de fordeler som oppnås gjennom statistisk multipleksing. I det først tilfellet, er gevinsten mulig som en følge av at forbindelsene ikke utnytter båndbredden. I det siste tilfellet er det mulig med en gevinst, på grunn av at flere kilder blir multiplekset sammen.

Det er to måter å stille inn CAC funksjonen på [26]:

1. Legge på en overbookings faktor i beregningen av den effektive båndbredden.

2. Skalere (bruke skaleringsfaktorer) trafikk deskriptorene før en legger på beregningen for å utlede den effektive båndbredden.

Siden sammenhengen mellom trafikk deskriptorene og den allokerte båndbredden ikke er lineær, fører bruke av booking og skaleringsfaktorer til forskjellige nivåer av overbooking. I begge tilfellene, hvis de ikke er nøyaktig utviklet, kan overbooking resultere i at QoS målene ikke overholdes for alle forbindelsen som deles av den overbookede ressursen.

Skalerings faktoren blir gitt ved å måle bruken av båndbredden over lang tid, for hver forbindelse. Hvis det blir fastslått at forbindelser med gitt spennvidde av trafikk deskriptorer, eller trafikk som kommer fra en viss type applikasjoner, ikke bruker mer en 50% av SCR. Da kan SCR bli skalert ned med max 50%, og gir rom for feil og uforutsette tendenser. Verdien for nedskalering blir brukt til CAC formål, siden forbindelsen skal kunne generere skurer etter de forhandlede trafikk deskriptorene. Hvis ytterligere målinger indikerer at bruken vokser eller avtar, kan skalerings faktorene modifiseres tilsvarende.

Booking faktorer blir gitt ved å måle hvordan køene vokser. Målinger av metning er ofte tilgjengelig for å gi innsikt med hensyn til om køen kan overbookes, uten å påvirke tjenestekvaliteten. Målinger av metning indikerer om køen har vokst over spesifiserte grenser for en gitt

periode. Basert på denne informasjonen, er det mulig å bestemme hvor aggressivt ressursen kan overbookes.

Det er vanskelig å gi en definitiv framgangsmåte på hvordan en skal stille inn CAC best mulig, for å oppnå meget høy utnyttelse ved å bare bruke trafikk med statistisk allokert båndbredde. Dynamikken til et SCR basert nettverk gjengir oppgaven mer kompleks, fordi den målte statistikken varierer sannsynlig vis med miksen av forbindelser til en gitt tid. Det er viktig notere seg at allokerte ressurser, som ikke er utnyttet, kan til en hver til bli effektivt brukt av BoD tjenester (ABR, GFR, UBR). Derfor kan det totale nettverket bli effektivt utnyttet.

6.3 Trafikk tilpasning

Som diskutert tidligere, forhandler applikasjonene fram QoS og båndbredde kravene gjennom trafikk kontrakten. Connection admission controll (CAC) allokerer tilstrekkelig båndbredde til forbindelsen, for å møte disse kravene.

Imidlertid er allokering av ressurser gjennom CAC ikke nok for å sikre at QoS blir holdt. Det vil si, at en forbindelse kan (med vilje eller tilfeldig) overskride de stadfestede trafikk deskriptorene og dermed få redusert QoS [26]. Denne forbindelsen kan også få betydning for tjenestekvaliteten til andre forbindelser. Siden nettverket er nødt til å møte QoS garantiene, til i hvert fall de cellene som er tilpasset, er det kritisk at en har muligheten til sikre at forbindelsen følger den fastsatte trafikk kontrakten. Denne muligheten får man gjennom implementering av *trafikk forming*, *trafikk policing* og *soft policing* mekanismer.

Det er usannsynlig at trafikken fra en applikasjon naturlig oppfører seg etter trafikk deskriptorene. For å unngå en reduksjon av QoS, må forbindelsen (ved ATM grensesnittet) forme trafikken for å sikre at den følger de fastsatte trafikk deskriptorene. Selv om standardene [27] og [28] ikke krever at trafikk forming skal være implementert, må applikasjonene implementere trafikk forming, for å tilpasse seg til de forhandlede trafikk deskriptorene. Ellers vil nettverket ta drastiske virkemidler i bruk på trafikken som ikke er tilpasset.

Et nettverk som forplikter seg til en QoS kan ikke bare stole på at forbindelsen føyer seg etter trafikk deskriptorene. Nettverket kan derfor overvåke hver forbindelse for å sikre at celler tilpasser seg til de fastsatte trafikk deskriptorene og gripe inn på de cellene som ikke innordner seg, for å sikre at de ikke påvirker QoS. Denne overvåkingen blir vanligvis gjort ved inngangen til nettverket og på grensene mellom forskjellige nettverk.

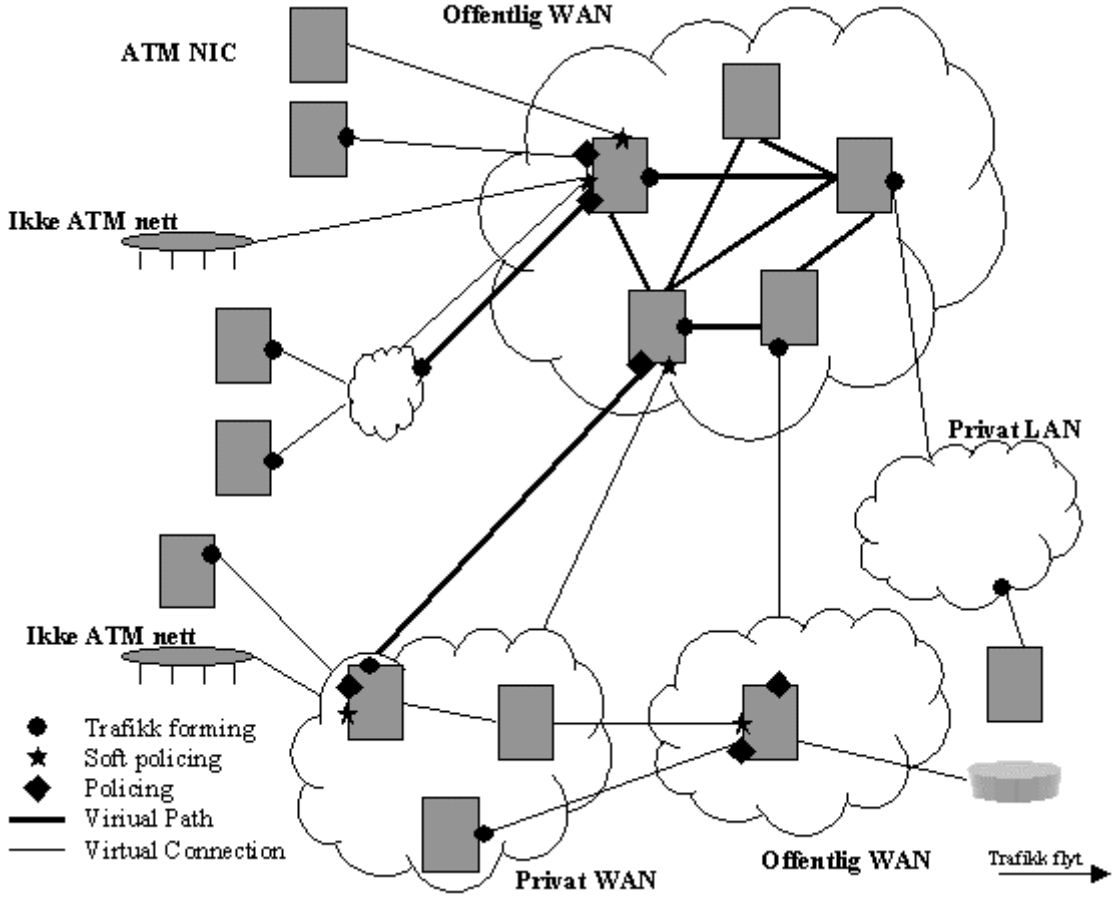
For en gitt forbindelse, blir en celle klassifisert som tilpasset eller ikke tilpasset til en trafikk deskriptor, gjennom en applikasjon som benytter en algoritme som kalles *generic cell rate algorithm* (GCRA) [26]. GCRA definerer hva som menes med å tilpasse seg til en gitt sett med trafikk deskriptorer.

Nettverket kan implementere GCRA for å overvåke trafikken og gripe inn på noen av cellene som ikke er tilpasset. Når nettverket finner en celle som ikke er tilpasset, kan den *merke* eller *forkaste*, eller den kan slippe den inn i nettverket som om den var tilpasset. Med *merking* mener en at en degraderer prioriteten til cellen (fra CLP = 0 til CLP = 1). Når en senker prioriteten til en celle, vil den ikke får noen QoS garantier, men den kan likevel kommer fram til destinasjonen hvis nettet ikke er overbelastet. Med *forkasting* mener en ganske enkelt at en fjerner

pakken fra trafikk strømmen. Alle cellene med CLP = 1, der det blir oppdaget at den ikke er tilpasset, kan ikke degraderes til en lavere prioritet og kan bli forkastet. Alternativt, kan nettverket implementere en formingsfunksjon, som sikrer at trafikken som kommer inn i nettet, tilpasser seg til trafikk deskriptorene, men som også tillater noe buffring av trafikken, som ikke er tilpasset, helt til trafikken kan slippes inn i nettet som tilpasset [26]. Denne metoden kalles *soft policing*.

For å forhindre at cellene blir merket eller forkastet, utfører trafikk kilden en trafikk formingsfunksjon. Trafikk formeren forsinker sendingen av cellen til den er kvalifisert som tilpasset av GCRA.
Hvis kilden former trafikken etter GCRA og hvis nettverker overvåker den, vil ingen celler bli berørt. Hvis kilden ikke kan forme trafikken, kan nettverket forme den i steden for å overvåke den.

Figur 23 viser de forskjellige lokasjonene hvor trafikk forming og trafikk overvåkning funksjoner kan være plassert i nettverket.



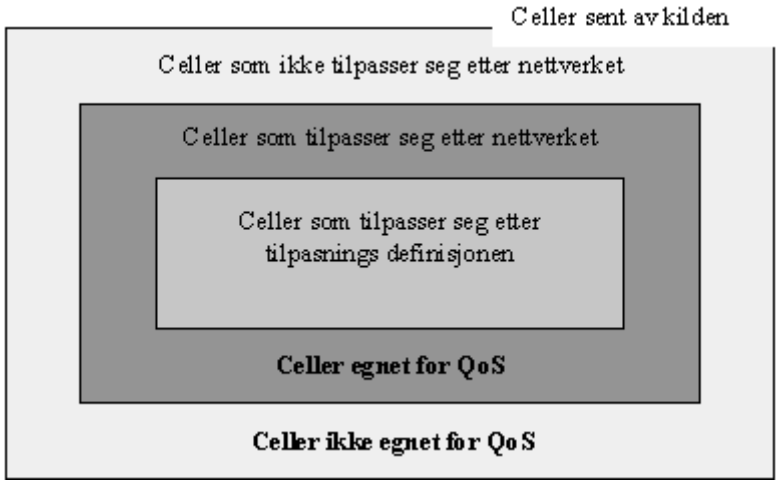
Figur 23: Hvor i nettet trafikk forming og trafikk policing kan være plassert.

6.3.1 Definisjon av tilpassing

Tilpasningsdefinisjonen bestemmer hvilke type celler (enten CLP = 0 eller CLP = 0 + 1) som QoS og trafikk deskriptorene er definert, og hvilke handlinger nettverket kan ta på trafikk som ikke er tilpasset [26]. Tilpasningsdefinisjonen er inkludert som en del av trafikk kontrakten.

Siden CLP = 1 celler har lavere prioritet og generelt ikke har noen QoS garantier, blir det ikke gitt noen tilpasnings definisjon på dem, hvis de ikke inkludert i den samlede flyten. Tilpasnings definisjonen pålegger ingen handlinger fra nettverket, men definerer området med tilpassede celler som er egnet for å motta den forhandlede QoS. Nettverket må kunne gi denne QoS, til minst denne mengde med tilpassede celler. Figur 24 viser tilpasnings området som sammenhengen med QoS forpliktelser.

Når QoS garantien blir gitt til den samlede trafikken (CLP = 0 + 1), bli CLP = 0 og CLP = 1 cellene behandlet likt, uten differensiert prioritet. Tilpasnings definisjonen er derfor CLP transparent.



Figur 24: Tilpasningsområdet

Det kan være mer enn en tilpasnings definisjon for en gitt tjeneste kategori, for å tillate en fleksibilitet når en designer tjenestetilbud. Definisjonen indikerer også at hvilken handling nettverket kan ta på celler som ikke er tilpasset. Som nevnt tidligere er disse handlingene enten å merke, forkaste eller la cellen passere som tilpasset. Hvis cellen kommer inn i nettet som tilpasset, er den egnet for QoS (f.eks. at nok ressurser må allokeres for alle tilpassede celler som kommer inn i nettet). Merking gjelder bare CLP = 0 flyten. PCR blir alltid definert på den samlede trafikken (CLP = 0 + 1). Derfor kan en kilde aldri sende noen celler (uavhengig av CLP biten) med en rate høyere enn PCR.

6.3.2 Trafikk overvåkning

Nettverket må sikre at den innkommende trafikken er tilpasset, fordi den allokerer båndbredde og garanterer QoS til tilpassede celler. Nettverket griper inn på celler som ikke er tilpasset, ved å enable en overvåkingsfunksjon for VC [26]. Overvåkning blir generelt satt i stand ved bruker-til-nettverks grensesnittet. Denne funksjonen blir kalt *user parameter control* (UPC). Overvåking kan bli satt i stand mellom to nettverk og blir da kalt *network parameter control* (NPC). Trafikk trenger bare å overvåkes ved inngangen til nettet. Overvåknings funksjonen implementerer en algoritme som er lik GCRA algoritmene.

Overvåknings funksjonen er ikke påtrengende, og verken forsinker eller modifiserer karakteristikken til celle flyten, annet en ved å droppe celler eller å senke prioriteten ved å sette CLP biten til 1.

Implementeringen av overvåkingsfunksjoner er ikke standardisert. En implementasjon som finner flere celler som ikke er tilpasset enn GCRA er ikke akseptabel, fordi den vil få innflytelse på den garanterte QoS. En ”snill” algoritme, som tillater at ikke tilpassede celler kan komme inn i nettet som tilpassede, er akseptabel, men da må nettverket garantere deres QoS og allokere flere ressurser.

Implementasjoner kan bare være i stand til overvåke et forhåndsbestemt sett med rater. Når en forbindelse blir satt opp, blir dens trafikk deskriptorer satt til det neste høyere nivået som støttes. For eksempel hvis overvåkningsfunksjonen bare støtter inkrementer av 10Mbps og en forbindelse ønsker en rate på 11Mbps, blir denne forbindelsen overvåket ved 20Mbps. Oppløsningen til overvåknings funksjonen påvirker direkte effektiviteten til nettverket, fordi verdiene til trafikk deskriptorene, som blir brukt til å allokere båndbredde i CAC, er verdiene som blir brukt i overvåkningen. CAC må ta i betraktning at en grådig bruker kan fylle all den ekstra båndbredden som tillates av overvåkningsfunksjonen med data, som nettverket betrakter som tilpasset og klar til å motta QoS. Derfor vil 11Mbps forbindelsen som ble nevnt tidligere, bli allokert 20Mbps. Det vil si en sløsing av båndbredde på 9Mbps. Hvis oppløsningen er mindre, vil allokasjonen av båndbredden være mer effektiv. ITU-T har standardisert den nødvendige oppløsningen for å kunne gi alle link rater lik presisjon. Overvåknings funksjonen må kunne håndtere en celle rate på 1% større enn den fastsatte raten [26].

6.3.3 Trafikk forming

Kilden, tilpasnings enheten, eller nettverket kan bruke en trafikk formings funksjon til en VC eller VP. Denne funksjonen modifiserer trafikk karakteristikken, for å tilpasse den til de fastsatte trafikk deskriptorene. Selv om trafikk forming er en valgfri trafikk håndterings mulighet, er den ett grunnlag for effektiv bruk av trafikk ressursene. Som beskrevet tidligere kan nettverket foreta noen drastiske handlinger på den trafikken som ikke er tilpasset, som kan føre til retransmisjon av informasjon av høyere lags protokoller som igjen fører til ueffektiv bruk av ressursene. Det er lite sannsynlig, hvis ikke umulig, at trafikk fra en applikasjon, på en magisk måte, tilpasser seg et sett med definerte trafikk deskriptorer som SCR og PCR. For å kunne lage en tilpasset celle flyt, må enheten som lager cellene forme trafikken [26].

Målet med trafikk forming er å lage en celle flyt som tilpasser seg trafikk deskriptorene, som passer tilpassningsalgoritmen som beskrevet. Den formede celle strømmen bør ha minimal CDV [26].

Som demonstrert i figur 19, er effekten av jitter i en multiplekser svært viktig. Selv hvis en kilde former trafikken, kan noen celler bli kategorisert som ikke tilpasset, på grunn av at celle strømmen ble utsatt for jitter. I noen tilfeller, på grunn av dette, er det nødvendig å ha muligheten til å omforme trafikken før den kommer inn i et nett som monitorerer trafikken.

Forming er også viktig for håndtering av VP’er. Det er umulig å karakterisere, a priori trafikk karakteristikken, som oppstår når en samler flere VC’er på en VP. Etterhvert som VC’er blir lagt til eller fjernet fra VP, blir karakteristikken også modifisert. Den resulterende trafikk karakteristikken kan være skuraktig eller jevn. VP blir laget med spesifikke trafikk deskriptorer, og den eneste måten den kan tilpasse seg til disse er gjennom trafikk forming.

For å illustrere kompleksiteten, kan en anta en PCR på 0,10 av linje raten, SCR på 0,05 av linje raten og MBS på 10 celler for hver av de seks VC'ene som blir samlet på en VBR VP. Det er praktisk umulig å karakterisere oppførselen til den resulterende strømmen. I verste tilfellet, kan en anta VBR VP'ens PCR vil være lik linje raten, at MBS vil være 60 celler og at SCR er summen av de individuelle SCR'ene (f.eks. 0,30 av linje raten). SCR antagelsen er den mest realistiske. PCR og MBS er "worst-case" antakelser og fører til at en ikke utnytter VP, siden cellene til hver VC ikke nødvendigvis ankommer likt. Den nedre grensen på PCR er 0,60 av linje raten, og den nedre grensen på MBS er 1. Hvis 10 ytterligere VC blir satt opp, vil ikke den orginale antagelsen holde. Siden VP blir bestemt tidligere uten noen kunnskap om blandingen av VC'er som samles på den, er det viktig å sette opp en effektiv kombinasjon av trafikk deskriptorer for VP og bruke trafikk forming for å sikre at tilpasning til enhver tid.
Trafikk forming kan også utføres ved utgangen av nettverket, før den leverer cellen til ende stasjonen. Nettverkstilbyderen kan da garantere minimum jitter i celle strømmen, før cellen kommer inn i nettet til kunden, noe som vil kreve mindre buffring i ende systemet.

Formings funksjonen krever at trafikken køes på VC/VP basis og tillater celle transmisjon bare når den føyer seg etter GCRA funksjonen med ingen CDVT. På grunn av buffring og omplan-legging av celler, kan formings funksjonen øke ende-til-ende forsinkelsen. Ved å legge på formingsfunksjonen igjen, innen et nettverk, der trafikken tidligere har vert formet, bør bare øke forsinkelsen en tanke, siden trafikken allerede oppfører seg riktig.

Målet med en trafikk former er minimalisere den målte Max-CDV ved dens utgang. Fordi mange VC'er kan kjempe om den samme linken og blir prosessert av den samme formeren, er det en sjanse for at celler kan sendes samtidig (kollisjon av celler). Formings funksjonen sender da cellene så fort som mulig, tatt i betraktning forsinkelses kravene til forbindelsen. Kollis-jonene forårsaker jitter i det formede trafikk mønsteret, derfor, selv om trafikken er formet, kreves det en CDVT $\neq 0$ i en nedstrøms overvåknings funksjon for å beregne for kollisjon.

Trafikk formingsfunksjonen sender aldri celler for tidlig, men når det oppstår kollisjon der cellene blir sendt for sent, blir det neste tidspunktet som en tilpasset celle blir sendt på, beregnet ut fra tiden den skulle ha blitt sendt og ikke på den effektive transmisjons tiden. Kollisjonen kan derfor danne en skur effekt, og formings funksjonen tillater kilden å få den fastsatte bånd-bredden. Ellers ville kollisjonen kunne føre til en reduksjon av båndbredden over lang tid av.

7 ATM og IP design

Med de QoS observasjoner som er gjort i det foregående kapittelet, blir neste spørsmål *"Hva slags QoS kan spesielt bli gitt av ATM i et heterogent nettverk som Internet, som delvis bruker ATM på transport nivå?"*

Når en betrakter dette spørsmålet, kommer de grunnleggende forskjellene i designet av ATM og IP til syne. Den fundamentale filosofien i Internet, er å gi ende-til-ende overførings tjenester, som ikke er avhengig av en bestemt transport teknologi og som fungerer langs en bane som bruker mange forskjellige transport teknologier. For å oppnå dette, bruker signaleringsmekanismen i TCP/IP to grunnleggende parametere for å karakterisere trafikken ende-til-ende [3]:

- *Et dynamisk estimat av ende-til-ende RTT* (Round Trip Time)
- *Tap av pakker*

Hvis nettverket fremstiller en oppførsel, der det oppstår metning innen et vindu i RTT, kan ende-til-ende signalering nøyaktig oppdage og justere den dynamiske oppførselen til nettverket.

ATM, som mange andre linklag transport teknologier, bruker et større sett med signaleringsmekanismer. Intensjonen her er å støtte et større sett med transport applikasjoner, samt forskjellige real-time applikasjoner og tradisjonelle ikke real-time applikasjoner. De større signaleringsmulighetene er tilgjengelige, ganske enkelt på grunn av den homogene naturen til ATM nettverket. Disse signaleringsmulighetene kan bli brukt til å støtte et stort utvalg av trafikk formings profiler, som er tilgjengelig i ATM svitsjer. Imidlertid kan dette større signaleringsmiljøet, sammen med bruk av en profil tilpasset real-time trafikk med lav jitter toleranse, lage et noe annerledes metningsmønster. Real-time trafikk reagerer på metning, ved å redusere belastningen, på bakgrunn av at køing av data dramatisk kan øke jitteren og forlenge varigheten til metningen. Design målet i et real-time miljø, er å hurtig forkaste celler, for å fjerne metningstilstanden [20].

Resultatet av dette design målet, er at metningstilstander i ATM generelt blir fjernet godt innenfor tidsintervallet av en ende-til-ende RTT i IP [26]. Når ATM svitsjen forkaster pakker for å redusere køen, tar det opptil en RTT å signalere tap av pakker i IP. Når TCP reduserer vindusstørrelsen, som en følge av signaleringen, er metningstilstanden i ATM fjernet. Denne observasjonen viser at det er en utfordring å definere ATM sine trafikk formings karakteristikker for trafikk baner, som strekker seg over IP og ATM, for at ende-til-ende TCP sesjoner skal holde maksimal transmisjonsrate.

QoS målet for nettverk, som av natur er lik Internet, ligger prinsipielt å styre nettverket til å forandre oppførselen til IP laget, slik at visse IP pakker blir forsinket eller forkastet ved starten på en metningstilstand, for å forsinke eller unngå innflytelsen av metning på andre klasser med IP trafikk. Når en ser på IP over ATM, er problemet at det ikke finnes noen mekanisme for å mappe slike IP lags direktiver til ATM laget. Det er heller ikke ønskelig, gitt den lille celle størrelsen til ATM og kravet til rask prosessering og forkasting.

Det ser ut til at IP sin QoS tilnærming vil fungere best i IP over ATM. Hvis ATM nettverk er tilstrekkelig dimensjonert for å håndtere burst belastning, uten å måtte påta seg stor-skala *congestion avoidance* ved ATM laget, er det ikke behov for at IP laget skal påkalle mekanismer

for metnings håndtering. På denne måten blir det snarere et spørsmål å øke kapasiteten, en QoS innen ATM.

IP-baserte applikasjoner vokser konstant, så nettverkstilbydere som gir ATM må kunne bære IP trafikk effektivt. Det har vært en intens debatt rundt hva som er den beste måten å bære IP trafikk over et ATM nettverk. Det har blitt utført en mengde med studier, som sammenlikner TCP over ABR, UBR og GFR. Ytelsen er avhengig av konfigurasjonen av IW enhetene i nodene, med tanke på buffer og metningskontroll mekanismer [30].

Simulering av et bestemt nettverk og dets muligheter er nødvendig for å gjøre det rette valget. Imidlertid, kan noen observasjoner på høyere nivå, bli brukt i beslutningsprosessen. Bruken av ABR kan øke kompleksiteten til ATM nettet, men nettverket bærer bare informasjon som er brukenes for destinasjonen, mens en minimaliserer kravene til buffring i ATM nettverk. IP/ATM IW krever fortsatt store buffere, for å kunne håndtere potensiell metning i nettverket.

Ved bruk av UBR med *frame discard* eller GFR, kan en oppnå ytelser tilsvarende et IP nettverk, uten at IP/ATM IW må støtte store buffere. I dette tilfellet blir båndbredden oppstrøms fra metningspunktet brukt til å bære pakker som må retransmiteres. Hvis båndbredden er dyr, kan bruken av ABR gi fordeler som overveier den økte kompleksiteten og kosten til den økte bufferen i IW enheten.

For å gi bedre støtte for IP applikasjoner, må ATM nettverk implementere metningskontroll og *frame discard*.

IP beveger seg mot støtte av garantert QoS, og er designer derfor trafikk håndteringsmetoder lik de som er definert i ATM (f.eks. overvåkning, CAC, flyt kontroll). IP/ATM IW funksjonen vil mappe QoS i IP over til tjenestekategoriene i ATM.

8 Testing

Ettersom Cisco Systems er leverandør av nettverks produkter til Statoil, og har en nettverkslab tilgjengelig i Oslo. Ble det bestemt at det ville være hensiktsmessig å utføre utprøving av IPv6 og ATM hos Cisco Systems i Oslo. Cisco Systems stilte sin systemingeniør, Peter Gustafsson, til disposisjon hele uke 20.

8.1 Testscenarier

Fra begynnelsen var det meningen at en skulle utføre testene på IPv6. I den sammenheng ble det gjort en del forberedelser, for å være best mulig forberedt til testingen hos Cisco Systems. Det ble blant annet gjort et søk etter hvordan en kan implementere IPv6 i ende systemene, samt etter hvilke applikasjoner som støttet IPv6.

Ettersom Statoil vil integrere data, tale, video og telemetri inn i et nettverk, ville det vært hensiktsmessig å teste ut denne sammensetningen i et IPv6 nett. Det ble derfor søkt etter IPv6 applikasjoner for disse tjenestene. Det viste seg at det var utviklet klart flere applikasjoner for operativsystemene Linux og FreeBSD, enn Windows. Selv for disse operativsystemene (OS), var det et begrenset utvalg av applikasjoner for de nevnte tjenestene. Resultatet av dette søket, var at det skulle satses på Linux som OS i ende systemene.

Det ble funnet følgende freeware applikasjoner med støtte for IPv6 på Internet for Linux:

- *VIC*, er en applikasjon for å overføre video [10].
- *VAT*, er en applikasjon for å overføre tale [10].
- *FTP*, er en applikasjon for filoverføring mellom to ende systemer.
- *Trafikk generator*, for å generere trafikk mellom to ende systemer [9].

For å spare tid inne i Oslo, ble det bestemt at det skulle installeres IPv6 på et par klienter som skulle tas med inn til Cisco. Det ble valgt å benytte en SUN Microsystems SPARC maskin, som skulle benyttes til å boote to diskløse klienter. Installasjonen av IPv6 gikk forholdsvis greit, men det viste seg at applikasjonen som ble funnet var ustabile eller manglet tilstrekkelig dokumentasjon.

Et av mine personlige delmål ved denne installasjonen, var også å få adgang til det såkalte 6Bone nettet, som er et verdensomspennende IPv6 testnett, der alle kan få tilgang. Sannsynligheten for at de applikasjoner som trengs i testen finnes der er stor. Jeg understreker at dette ikke var en del av målsetningen ved hovedoppgaven, men snarere av personlige interesser.

Etter forøket med installasjon av programvaren for IPv6 på Linux, ble det bestemt at testene skulle utføres på IPv4 i steden. Denne beslutningen ble tatt på grunnlag av den begrensede tiden som ble satt av til testing, samt at å utføre testen på IPv4 ikke ville skille seg nevneverdig ut fra en test på IPv6. Det ble i hovedsak utarbeidet to test scenarier.

8.1.1 LAN – WAN – LAN

I dette testscenariet skal to klienter på, hver sin side av en ATM svitsj og to IP svitsjer, kommunisere med hverandre. En skisse av dette nettverket kan ses i figur 25. Hensikten med denne testen er å se om QoS ivaretas, når en legger inn et ATM nett under IP nettet.

Følgende målinger skal utføres:

- Tap av pakker.
- Forsinkelse.
- Throughput.
- Kvaliteten på de forskjellige tjenestene.
- Hva som skjer ved metning i nettet.
- Installere RSVP og se på kvaliteten til de forskjellige tjenestene da.

Det som kan forventes av en slik test, er at throughput, d.v.s. den målte ytelsen til nettverket, vil være redusert i forhold til en ren LAN løsning. Dette kommer av at pakkene må segmenteres og reassembleres, før de kan sendes videre i IP nettet.

Figur 25: Skisse av LAN - WAN - LAN nettverk.

8.1.2 LAN

I dette testscenariet skal to klienter kommunisere med hverandre i en ren LAN løsning.

Figur 26: Skisse av LAN nettverk.

De samme målingene som i det foregående scenariet skal utføres her. Her vil en antagelig oppleve at throughput går opp, som en følge av at en slipper segmentering og reassemblering. Med dette slipper man også prosesseringstiden av denne mekanismen.

8.2 Mangler

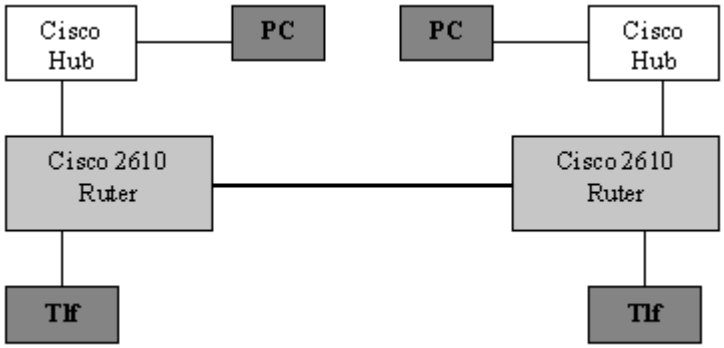
Testlaboratoriet ved Cisco i Oslo er rimelig nytt og ved ankomst Oslo viste det seg at mange av de applikasjonene som trengtes for å gjøre testen manglet. Jeg måtte da revurdere testscenariene

som er beskrevet over. Dette problemet ble det fra Ciscos side gjort oppmerksom på før avreise, og vi ble enige om at testscenariene skulle revurderes hos Cisco i Oslo.

I samarbeid med Peter Gustafsson kom vi fram til at vi kunne utføre målinger av throughput mellom to klienter, med testoppsettet beskrevet i 7.1.1 og 7.1.2.

I tillegg kunne vi koble opp to telefoner tilknyttet en Cisco Hub og en Cisco 2610 ruter i hver ende. Samtidig kunne vi sende data mellom to Windows PC'er ved hjelp av FTP (se figur 27). Med overføring av både tale og data på samme linje kan en finne ut hvordan kvaliteten på talen blir.

I software i ruterene kan en bestemme hvor stor båndbredde serielinken mellom ruterene skal ha. Dermed kan en begrense båndbredden, slik at tale- og dataforbindelsen må kjempe om ressursene.



Figur 27: Test oppsett for RSVP.

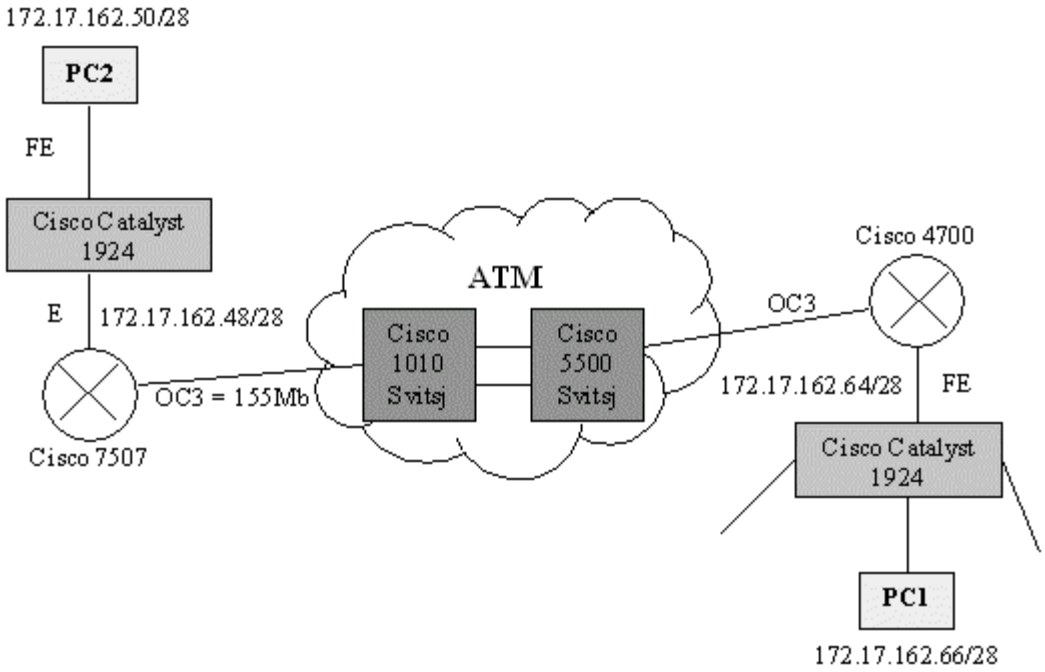
Etter å ha gjort dette innledende forsøket, installerte vi RSVP slik at vi kunne reservere en viss båndbredde for tale forbindelsen. Med dette oppsette kan vi få testet hvordan RSVP fungerer og om det er lønnsomt å satse på dette.

8.3 Resultater

Vi koblet opp nettverket, som vist i figur 28, for å måle throughput til forskjellig nettverks-løsninger. Til målingene brukte vi et program, som genererte pakker mot det andre ende systemet. I dette programmet kunne en velge om en skulle sende TCP eller UDP pakker. Programmet genererte også en fil der all informasjonen om forbindelsen ble lagret, deriblant throughput.

Grensesnitt	Pakketype	Nettverk	Throughput
10 – 10 Mbps	TCP	LAN	8,55 Mbps
10 – 10 Mbps	UDP	LAN	9,50 Mbps
100 – 10 Mbps	TCP	ATM	7,20 Mbps
100 – 10 Mbps	UDP	ATM	8,90 Mbps
10 – 10 Mbps	TCP	ATM	6,50 Mbps
10 – 10 Mbps	UDP	ATM	9,50 Mbps

Tabell 2: Måleresultater av throughput



Figur 28: Testoppsett for måling av throughput.

I tillegg til å forandre hvilke type pakker som ble sendt, konfigurerte vi også nettverkskortene i PC'ene til å bruker 10 og 100Mbps grensesnitt. Vi valgte dette, for å se hvor stor innvirkning dette hadde på throughput. Resultatet av målingene kan ses i tabell 2.

Hvis vi først tar for oss den rene LAN løsningen, ser vi at vi får en større throughput for UDP enn TCP. Dette er ikke så overraskende ettersom UDP har en enklere header enn TCP. Det vil si at det er mindre header data, i forhold til brukerdata. Det som er noe overraskende, er at throughput, i begge tilfeller, er så høy.

Ser vi på LAN kontra WAN (d.v.s. ATM) løsningen, ser vi at throughput er generelt lavere for WAN løsningen. ATM pakkene er relativt små (53 bytes) i forhold til TCP og UDP. Derfor må TCP og UDP pakkene segmenters når de kommer ned til ATM laget. I motsatt ende, må de re-assembleres igjen. Denne prosesseringen gjør at ytelsen til nettverket synker, noe som vises i tabell 2.

Båndbredde	RSVP	Kbps reservert	Antall FTP sesjoner	Kvalitet på talen
2Mbps	Nei	Ingen	1	Meget bra lydkvalitet
1,3Mbps	Nei	Ingen	1	Meget bra lydkvalitet
500Kbps	Nei	Ingen	1	Meget bra lydkvalitet, men litt forsinkelse
148Kbps	Nei	Ingen	1	Hakking og stor forsinkelse
128Kbps	Nei	Ingen	1	Umulig å oppfatte hva motparten sier
64Kbps	Nei	Ingen	1	Umulig å oppfatte hva motparten sier
500Kbps	Nei	Ingen	2	Umulig å oppfatte hva motparten sier
128Kbps	Ja	96Kbps	2	Bra lydkvalitet, men litt forsinkelse
128Kbps	Ja	48Kbps	2	Bra lydkvalitet, men litt forsinkelse
64Kbps	Ja	48Kbps	2	Umulig å oppfatte hva motparten sier
64Kbps	Ja	10Kbps	2	Umulig å oppfatte hva motparten sier
64Kbps	Ja	10Kbps	1	Meget bra lydkvalitet

Tabell 3: Måleresultaer av RSVP.

I det andre eksperimentet, der vi testet hvordan en tale forbindelse ble påvirket av en FTP forbindelse på samme link, fikk vi resultatene i tabell 3.

Som vi ser av tabellen, er det fordelaktig å benytte seg av RSVP, hvis en har begrenset båndbredde (noe man som regel har). Hvis en hadde en tilgjengelig båndbredde på 64Kbps og reserverte 10Kbps til tale, samtidig som en hadde en FTP forbindelse oppe, fikk en en meget god lydkvalitet. Hvis en ikke benyttet seg av RSVP, måtte en ha en tilgjengelig båndbredde på 500Kbps for å få en tilnærmet kvalitet. Dette betyr at en kan få en grei forbindelse for data og lyd med ISDN, der en B-kanal er på 64Kbps, ved å benytte RSVP.

Ved å øke antall FTP forbindelser måtte en imidlertid doble båndbredden for å få en tilnærmet lydkvalitet, ved bruk av RSVP. En kan si at RSVP brøt sammen, når båndbredden ble strupet. Dette kan skyldes flere årsaker. En av de kan være at RSVP, utilsiktet, inngår et kompromiss ved metning, der den reduserer den reserverte båndbredden.

Jeg la også merke til at telefonlinjen virket ”død”, når ingen snakket. Dette skyldes at det ikke blir sendt noe data, når det ikke er noe å sende. Derfor høres det ”dødt” ut.

8.3.1 Feilkilder

Ettersom disse målingene ble gjort i et test miljø, og ikke i et virkelig system, kan det oppstå feilkilder.

- Vanligvis vil det være større belastning i nettet, enn det vi klarte å generere. Dette kan få innvirkning på testresultatene.

- Det programmet jeg benyttet for å måle throughput, kan måle feil. Dette er et program som Cisco har benyttet tidligere til målinger og har ikke oppdaget noe feil med det. Selv om programmet kan likevel være en feilkilde.

9 Diskusjon og konklusjon

Det er en uenighet om nødvendigheten av IPv6, og når en gransker de punkter som blir brukt av begge parter i denne diskusjonen, er det vanskelig å finne grunner for en massiv implementering av IPv6, i hvertfall på kort sikt. Faktisk har de fleste egenskapene som er lagt til i IPv6, også blitt implementert i IPv4.

Traffic Class feltet i IPv6 gir ingen store forbedringer i forhold til IP prioritets feltet i IPv4. Ingen har hittil klart å kommersielt implementere differensierte tjenesteklasser, og det er derfor ingen grunn til å oppgradere til IPv6 av den grunnen.

Anvendelsen av IPv6 sin Flow Label, kan vise seg å være svært nyttig. Den eneste merkbare bruken av IPv6 sin Flow Label er i sammenheng med ressursreserverings protokoller, som f.eks. RSVP. *Der kan Flow Label feltet bli brukt til å gi en bestemt flyt, en spesifikk reservasjon.* Tilstedeværelsen av Flow Label feltet i pakker, kan benyttes for å sende pakker gjennom mellomliggende noder, som tidligere har etablert en bane eller reservasjonstilstand, for bestemte flyter. Bortsett fra bruken av Flow Label i RSVP, er fordelene med dette feltet uforutsigbar.

Ved valg av Ethernet som linklagsbærer, vil en ikke få noen garantier for QoS, utover "best effort". For at dette skal muliggjøres, må det implementeres tilleggsmekanismer (f.eks. RSVP), for å oppnå garantert QoS.

Alt i alt kan en si at IPv6 ikke gir QoS forbedringer, utover det som allerede er mulig i IPv4.

Når det gjelder ATM, har denne teknologien mange egenskaper som gir mulighet for QoS, som tillater virtual connections med konstant eller variabel bitrate, *burst* muligheter og trafikk forming. Selv om ATM har mange QoS muligheter, er ikke teknologien allment brukt som en ende-til-ende transport protokoll.

På bakgrunn av dette, er ikke ATM ideell til bruk i innføring av QoS. Selv om det finnes situasjoner der ende-til-ende banen gjennomtrengende består av ATM, er det sjelden at endesystemene er tilknyttet ATM. Hvis vi imidlertid ser isolert på ATM, kan teknologien gi QoS garantier. *ATM teknologien er unik på den måten den, på et nettverk, støtter flere applikasjoner som krever forskjellig QoS.*

Relasjonen mellom ATM og de to IP versjonene fortjener spesiell oppmerksomhet. IPv4 over ATM er kjent som klassisk IP over ATM. *IPv6 er ikke utviklet for å utnytte QoS og trafikk håndteringsmulighetene som finnes i ATM, spesielt bedre enn IPv4.* Begge teknologiene benytter ATM i første rekke som linklags teknologi.

Både IPv6 og ATM er teknologier som står sterk i dagens tele- og datakommunikasjonssamfunn. *IPv4 er den protokollen som blir mest benyttet i dagens nettverk og dens arvtager, IPv6, vil følgelig om noen år, overta IPv4 sin plass.* Det gjenstår en del standardiseringsarbeid, samt kommersielle applikasjoner som støtter IPv6, før IPv6 er moden for implementasjon.

Når det gjelder ATM, har denne teknologien etablert seg som den dominerende linklagsteknologien i dagens Internet. Dette kommer ganske enkelt av at det er den eneste teknologien som gir transporthastigheter av data over OC3 (155Mbps) p.d.d., samt fleksibiliteten når det gjelder multipleksing i ATM implementasjoner. *ATM's fremtid er høyst usikker, men min beskjedne mening er at ATM vil være med i mange år fremover.*

Mye arbeid og kartlegging gjenstår før en får svar på hvordan IP6 og ATM kan garantere tjenestekvalitet. Jeg tenker da først og fremst på hvordan sikkerheten ivaretas i disse teknologiene. Det vil også være fordelaktig å gjøre en større simulering, for å se hvordan en integrert løsning av tjenestene data, tale, video og telemetri fungerer, før en slik implementasjon skjer.

Litteraturreferanser

[1] Lars Line
***IT 6401 Hovedoppgave**, Høgskolen i Agder (1999)*
http://fag.grm.hia.no/it6401

[2] Bodil Johansen
***Virtual Private Network: Protokoller og sikkerhetsmekanismer,** Hovedoppgave ved Sivilingeniørutdanningen i Grimstad våren 1999.*

[3] Larry L. Peterson & Bruce S. Davie
***Computer Networks: A Systems Approach**, Morgan Kaufmann Publishers, Inc (1996)*

[4] Christian Huitema
***IPv6: The Internet Protocol**, Prentice Hal, Inc (1996)*

[5] S Dearing & R. Hinden
***Internet Protocol, Version 6 (IPv6) Specification**, RFC 2460, IEFT Network Working Group (Desember 1998),*
ftp://ftp.isi.edu/in-notes/rfc2460.txt

[6] S. Shenker & C. Partridge & R. Guerin
***Specification of Guaranteed Quality of Service**, RFC 2212, IETF Network Working Group (September 1997),*
ftp://ftp.isi.edu/in-notes/rfc2212.txt

[7] Paul Ferguson & Geoff Huston
***Quality of Service, Deliviering QoS on the Internet and in Corporate Networks**, Wiley Computer Publishing (1998)*

[8] Bay Networks, Inc
***The Case for IPv6**, White Paper, Bay Networks, Inc (1997),*
http://dcn.soongsil.ac.kr/~yskim/research/2789.html

[9] Paul E. McKenny, Danny Y. Lee, Barbara A . Denny
***Traffic Generator Software Release Notes**, Information and Telecommunication Science Center (November1995)*

[10] Information and Computing sciences Division, Ernest Orlando Lawrence Berkeley National Laboratory
VIC & VAT Beginner's Guide
http://www-itg.lbl.gov/mbone/

[11] Richard P. Draves, Allison Mankin, Brian D. Zill
***Implementing IPv6 for Windows NT**, Microsoft Research (1998)*
http://www.research.microsoft.com/msripv6/usenixnt/paper.htm

[12] Ingo Busse, Bernd Deffner, Henning Schulzrinne
***Dynamic QoS Control of Multimedia Applications based on RTP**, GMD-Fokus (1995)*

[13] Stefan Schmid, Andrew Scott, David Hutchison, Konrad Froitzheim
***QoS based Real-Time Audio Streaming in IPv6 Networks**, Lancaster Universit, U.K., University of Ulm, Germany (1998)*

[14] Peter Newman, Tom Lyon, Greg Minshall
***Flow Labelled IP: Connectionless ATM under IP**, Ipsilon Networks Inc. (1996)*

[15] Kartik Gopalan, Anindya Neogi, Prashant Pradan, Srinidhi Varadarajan
***Rether: A Real-Time Ethernet Protocol**, Tzi-cker Chiueh Faculty,*
http://www.ecsl.cs.sunysb.edu/rether.html

[16] Alexey N. Kuznetsov
***Tunnels over IP in Linux 2.2**, Institute for Nuclear Research, Moscow (1999),*

[17] Alexey N. Kuznetsov
***IPv6 Flow Labels in Linux 2.2**, Institute for Nuclear Research, Moscow (1999),*

[18] S. Blake, D. Black, M. Carlson, E. Davies, m.f.
***An Architecture for Differentiated Services**, RFC 2475, IETF Network Working Group (December 1998),*
ftp://ftp.isi.edu/in-notes/rfc2475.txt

[19] J. Wroclawski
***The Use of RSVP with IETF Integrated Services**, RFC 2210, IETF Network Working Group (September 1997),*
ftp://ftp.isi.edu/in-notes/rfc2210.txt

[20] Lars Aarhus, Jannicke Riisnæs
***Emerging Network Protocols: From IPv6 and RSVP to ATM**, Norsk Regnesentral, Oslo (1998),*

[21] Eric C. Rosen, Arun Viswanathan, Ross Callon
***Multiprotocol Label Switching Architecture**, Internet Draft, IETF Network Working Group (1999),*
http://search.ietf.org/internet-drafts/draft-ietf-mpls-arch-05.txt

[22] R. Braden, L. Zhang, S. Berson, m.f.
***Resource ReSerVation Protocol (RSVP)**, RFC 2205, IETF Network Working Group (September 1997),*
ftp://ftp.isi.edu/in-notes/rfc2205.txt

[23] K. Nichols, S. Blake, F. Baker, D. Black
***Definition of the Differentiated Service Field (DS Field) in the IPv4 and IPv6 Headers**,*
RFC 2474, IETF Network Working Group (December 1998), ftp://ftp.isi.edu/in-notes/rfc2474.txt

[24] R. G. Cole, D. H. Shur, C. Villamizar
***IP over ATM: A Framework Document**, IETF Internet Draft, (February 1996),*
http://engr.ans.net/atm-framework/draft-ietf-ipatm-framework-doc-08.txt

[25] A. Makin
***Resource ReSerVation Protocol (RSVP) – Version 1 Applicability Statment – Some Guidelines on Deployment**,*
RFC 2208, IETF Network Working Group (1997)
ftp://ftp.isi.edu/in-notes/rfc2208.txt

[26] Natalie Giroux, Sudhakar Ganti
***Quality of Service in ATM networks**¸ Prentice HallInc (1999).*

[27] ATM Forum, Teknisk Komitee
***Traffic Management Specification**, version 4.0 (1996)*

[28] ITU-T-Telecommunication Standardization Sector
***Traffic Control and Congestion Control in B-ISDN**, recommendation I.371 (Mai 1996)*

[29] Martin Borriss
***QoS support in ATM and selected protocol implementations**, TU Dresden, IBDR (1995)*
http://www.inf.tu-dresden.de/~mb14/atm.html

[30] M. Laubach
***Classical IP and ARP over ATM**, RFC 1577, IETF Network Working Group (1994)*
http://www.iihe.ac.be/rfc/rfc1577.txt

[31] Kohei Shiomoto, Naoaki Yamanaka, Tatsuro Takahashi
***Overview Of Measurement-Based Connection Admission Control Methods in ATM Networks**, NTT Network Service Systems Laboratories (1999), http://www.comsoc.org/pubs/surveys/1q99issue/shiomoto.html*

[32]H. Schulzrinne, S. Casner, m.f.
***RTP: A Transport Protocol for Real-Time Applications**, RFC 1889, IETF Network Working Group (1996)*
http://ftp.isi.edu/in-notes/rfc1889.txt

Ordliste

AAL *ATM Adaptation Layer*. En forbindelse av protokoller som tar data trafikk og frames inn i en sekvens av 48 bytes payloader for transmisjon over et ATM nettverk.

ABR *Available Bit Rate*. En av tjenestekategoriene definert av ATM Forum. ABR støtter trafikk med variable bitrate med flyt kontroll.

ATM *Asynchronous Transfer Mode*. En forbindelsesorientert pakkesvitsjet transmisjonsarkitektur som har data celle på 53 bytes, som består av 5 bytes header og 48 bytes payload.

CAC *Connection Admission Control*. Definert som et sett med handlinger tatt av nettverket under oppkoblingen av forbindelsen, for å bestemme om forespørselen om adgang til forbindelsen kan aksepteres eller ikke.

CBR *Constant Bit Rate*. En tjenestekategori i ATM som samsvarer med en konstant båndbredde allokasjon for en trafikkflyt.

CDV *Cell Delay Variation*. En QoS parameter i ATM som måler variasjonen i transittiden til celler over en VC.

CDVT *Cell Delay Variation Tolerance*. Representerer grensen til graden av skurhet til cellene, som tolereres av overvåkningsfunksjonen i nettverket.

CER *Cell Error Ratio*. Representerer forholdet; celler med feil i en transmisjon med hensyn på antall sendte celler.

CET *Cell Emission Time*. Representerer avgangstiden til en celle ved en formingsfunksjon.

CLP *Cell Loss Priority*. Et felt i ATM headeren (1 bit), som indikerer prioriteten til cellen. CLP = 1 indikerer at cellen kan forkastes, hvis det oppstår metning.

CLR *Cell Loss Ratio*. Et ATM QoS mål, som er definert av forholdet, tapte celler på antall transmitterte celler.

CoS *Classes of Services*. En kategorisk metode å klassifisere trafikk inn i separate klasser, for å gi differensierte tjenester til hver klasse innen nettverket.

CRC *Cyclic Redundancy Check*. Matematisk algoritme som beregner en numerisk verdi basert på bitene i en data blokk. Dette nummeret blir transmittert med dataen, og mottakeren bruker denne verdien til å verifisere dataen, ved å sammenlikne verdien med resultatet av algoritmen.

CTD *Cell Transfer Delay*. Et ATM QoS mål, som måler transittiden til en celle over en VC.

FTP *File Transfer Protocol*. En TCP basert, transaksjonsorientert fil overførings protokoll brukt i TCP/IP nettverk, spesielt Internet.

GCRA *Generic Cell Rate Algorithm*. En spesifikasjon for implementasjon av celle-rate tilpasning for ATM VBR Virtual Connections. GCRA er en algoritme som bruker trafikk parametere for å karakterisere trafikk som er tilpasset til definerte tilgangs kriterier. GCRA implementasjonen blir vanligvis kalt *leaky bucket.*

HTTP *Hyper Text Transfer Protocol*. En TCP baset applikasjonslags protokoll brukt for å kommunisere mellom Web servere og Web klienter.

ICMP *Internet Control Message Protocol*. En nettverklagsprotokoll som gir tilbakemelding om feil og annen informasjon spesielt aktuell i håndtering av IP pakker.

IETF *Internet Engineering Task Force*. Et standardiseringsorgan for protokoller, som utvikler og standardiserer protokoller og Internet standarder, generelt i nettverkslaget og lavere.

IP *Internet Protocol*. Nettverklagsprotokollen i TCP/IP stakken som blir brukt i Internet.

IPng *IP Next Generation*. Et annet ord for arvtakeren til IP versjon 4, også kjent som IPv6.

IPv4 *Internet Protocol Version 4*. Den versjonen av Internet Protokollen som er mest utbredt i dag.

IPv6 *Internet Protocol Version 6*. Versjonsnummeret til den IETF standardiserte neste generasjons Internet Protokollen.

ISDN *Integrated Services Digital Network*. En tidlig adoptert protokoll modell, som i dag tilbys av mange telefonselskaper for digital ende-til-ende forbindelse.

ITU-T *International Telecommunication Union Telecommunications*. ITU-T er et internasjonalt organ med medlemsland, der oppgaven er å definere anbefalinger og standarder relatert til den internasjonale telekommunikasjonsindustrien.

IW *InterWorking*. En enhet som konverterer trafikk fra en protokoll til en annen (f.eks. IP til ATM)

Jitter En forvrengning av et signal som forplanter seg gjennom nettverket, hvor signalet varierer fra dens originale referansetid. I pakkesvitsjede nettverk, er jitter forvrengningen av tiden mellom pakkene, i forhold til det den ble sendt med.

LAN *Local Area Network*. Et lokalt kommunikasjonsmiljø, typisk konstruert ved bruk av private kapling og kommunikasjonsfasiliteter.

Leaky Bucket Generelt en trafikk formings mekanisme, der inngangssiden til formings funksjonen er av vilkårlig størrelse og utgangssiden har en mindre fast størrelse.

Max-CDV *Maximum Cell Delay Variance*. Er den absolutte verdien av maksimum en-punkts CDV målt.

Max-CTD *Maximum Cell Transfer Delay*. Er den $(1 - \alpha)$ kvantile til CTD som kan bli erfart av leverte celler på en forbindelse under hele holde-tiden til forbindelsen.

MBS *Maximum Burst Size.* Et ATM QoS mål, som beskriver det antall celler som kan transmitteres ved peak raten, samtidig som den holder seg innenfor GCRA terskelen til tjeneste kontrakten.

MCR *Minimum Cell Rate*. En ATM tjeneste parameter relatert til ABR tjenesten. Den tillatte celle raten kan variere mellom MCR og PCR for fortsatt være tilpasset til tjenesten.

MFS *Maximum Frame Size*. Er en del av GFR trafikk kontrakten og definerer den maksimalt tillatte frame størrelsen.

NNI *Network-to-Network Interface*. En ATM Forum standard som definerer grensesnittet mellom to ATM svitsjer, som opereres av den samme offentlige eller private nettverksoperatøren.

nrt-VBR *Non-Real-Time Variable Bit Rate.* En av to VBR tjenestekategorier, der timing informasjon ikke er avgjørende.

PCR *Peak Cell Rate*. En ATM tjeneste parameter. PCR er den maksimale verdien til transmisjonsraten til trafikk på en VC med tjenestekategorien ABR.

PDU *Protocol Data Unit.* Er en melding til en gitt protokoll, og omfatter payload og protokoll spesifikk kontroll informasjon, typisk plassert i headeren.

PTP-CDV *Peak-To-Peak CDV*. Er en QoS forsinkelsesparameter tilknyttet CBR og real-time VBR tjenester.

PVC *Permanent Virtual Connection*. En ende-til-ende VC, som er etablert permanent.

QoS *Quality of Service.* QoS er definert for hver forbindelse på en ende-til-ende basis, med hensyn på de følgende attributtene på en ende-til-ende ATM forbindelse: CLR, Max-CTD og PTP-CDV.

REM *Rate Envelope Multiplexing.* Er en tilgangs metode til forbindelsen, som gir adgang til forbindelser slik at den totale ankomst raten er mindre en link kapasiteten.

Ruting Prosessen med å kalkulere nettverkstopolog og banen informasjon basert på nettverklags informasjon i pakker.

RSVP *Resource ReSerVation Setup Protocol*. En IP basert protokoll, brukt til å kommunisere QoS krav til mellomliggende noder i nettverket.

RTT *Round Trip Time.* Tiden som blir brukt av data trafikk for å forplante seg fra sender til mottaker og tilbake igjen.

rt-VBR *Real-Time VBR*. En av to VBR ATM tjenestekategorier, hvor timing informasjon er avgjørende.

SAP *Service Advertisement Protocol.* En broadcast-basert, Novell NetWare protkoll, som blir brukt til å avertere tilgjengeligheten av individuelle applikasjonstjenester i et NetWare nettverk.

SCR *Sustained Cell Rate.* En trafikk parameter i ATM som spesifiserer den gjennomsnittlige raten som ATM celler kan transmitteres med over en VC.

TCP *Transmission Control Protocol.* TCP er en pålitelig, forbindelse- og byteorientert transportlags protokoll innen TCP/IP protokoll rekken.

Telnet En TCP basert terminal-emulasjons protokoll brukt i TCP/IP nettverk, hovedsaklig for å forbinde og logge seg på fjerne systemer.

Token Bucket En trafikk formingsmekanisme, der muligheten til å sende pakker fra en gitt flyt er kontrollert av tokener. For pakker som hører til en bestemt flyt skal transmitteres, må tokenet være til stede i bøtten.

TOS *Type Of Service.* Et felt i IP headeren, som er designet for å indikere hvordan pakker skal behandles i nettverket.

UBR *Unspecified Bit Rate.* En ATM tjenestekategori, som blir brukt til ”best effort” trafikk.

UNI *User-To-Network Interface.* Brukt til å vise til ATM Forum spesifikasjonen for ATM signalering mellom et bruker-basert enhet og en ATM svitsj.

VBR *Variable Bit Rate.* En ATM tjeneste karakterisering av trafikk som er skurete av natur eller variable i gjennomsnittet, peak og minimum rater som data blir transmittert med.

VC *Virtual Connection.* En ende-til-ende forbindelse mellom to enheter som spenner over lag 2.

VCI *Virtual Connection Identifier.* En numerisk identifikator brukt til å identifisere den lokal enden til en VC.

VP *Virtual Path*. En forbindelses bane mellom to ende systemer over et ATM svitsje struktur.

VPI *Virtual Path Identifier*. En numerisk identifikator brukt til å identifisere den lokal enden av en ATM VP.

VPN *Virtual Private Network*. Et nettverk diskret kan eksistere på en fysisk infrastruktur bestående av flere VPN’er.

WAN *Wide Area Network*. Et nettverksmiljø der elementene i nettverket er lokalisert langt fra hverandre, der kommunikasjonsfasilitetene er bærer fasiliteter, i steden for privat kabling.

WWW *World Wide Web*. En global samling av Web servere, sammenkoblet av Internet, som bruker HTTP.

Vedlegg

Kildefil ved måling av throughput

LAN – TCP – 10 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 15.46 real sec = 1059.63 KB/sec (8680489.46 bps)
wsttcp-t: 2048 I/O calls, msec/call = 7.73, calls/sec = 132.45
16777216 924346247.23 924346262.70 15.46 8680489.46

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 15.73 real sec = 1041.38 KB/sec (8530968.54 bps)
wsttcp-t: 2048 I/O calls, msec/call = 7.87, calls/sec = 130.17
16777216 924346529.84 924346545.57 15.73 8530968.54

LAN – UDP – 10 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 udp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: 16777216 bytes in 14.21 real sec = 1152.99 KB/sec (9445301.06 bps)
wsttcp-t: 2054 I/O calls, msec/call = 7.08, calls/sec = 144.55
16777216 924346585.90 924346600.11 14.21 9445301.06

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 udp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: 16777216 bytes in 14.09 real sec = 1162.73 KB/sec (9525067.63 bps)
wsttcp-t: 2054 I/O calls, msec/call = 7.02, calls/sec = 145.77
16777216 924346717.06 924346731.15 14.09 9525067.63

LAN – TCP – 100 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 18.77 real sec = 873.02 KB/sec (7151794.53 bps)
wsttcp-t: 2048 I/O calls, msec/call = 9.38, calls/sec = 109.13
16777216 924344583.65 924344602.41 18.77 7151794.53

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 18.54 real sec = 883.85 KB/sec (7240531.26 bps)
wsttcp-t: 2048 I/O calls, msec/call = 9.27, calls/sec = 110.48
16777216 924345042.25 924345060.78 18.54 7240531.26

ATM – TCP – 10 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 20.47 real sec = 800.39 KB/sec (6556801.56 bps)
wsttcp-t: 2048 I/O calls, msec/call = 10.23, calls/sec = 100.05
16777216 924345499.95 924345520.42 20.47 6556801.56

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 20.30 real sec = 807.13 KB/sec (6612036.45 bps)
wsttcp-t: 2048 I/O calls, msec/call = 10.15, calls/sec = 100.89
16777216 924345756.61 924345776.91 20.30 6612036.45

ATM – UDP – 10 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 udp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: 16777216 bytes in 14.02 real sec = 1168.62 KB/sec (9573304.42 bps)
wsttcp-t: 2054 I/O calls, msec/call = 6.99, calls/sec = 146.50
16777216 924345839.42 924345853.44 14.02 9573304.42

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 udp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: 16777216 bytes in 14.09 real sec = 1162.81 KB/sec (9525743.65 bps)
wsttcp-t: 2054 I/O calls, msec/call = 7.02, calls/sec = 145.78
16777216 924345974.58 924345988.67 14.09 9525743.65

ATM – TCP – 100 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 18.77 real sec = 873.02 KB/sec (7151794.53 bps)
wsttcp-t: 2048 I/O calls, msec/call = 9.38, calls/sec = 109.13
16777216 924344583.65 924344602.41 18.77 7151794.53

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 tcp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: connect
wsttcp-t: 16777216 bytes in 18.54 real sec = 883.85 KB/sec (7240531.26 bps)
wsttcp-t: 2048 I/O calls, msec/call = 9.27, calls/sec = 110.48
16777216 924345042.25 924345060.78 18.54 7240531.26

ATM – UDP – 100 – 10

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 udp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: 16777216 bytes in 1.51 real sec = 10835.98 KB/sec (88768338.62 bps)
wsttcp-t: 2054 I/O calls, msec/call = 0.75, calls/sec = 1358.47
16777216 924344692.32 924344693.83 1.51 88768338.62

wsttcp-t: buflen=8192, nbuf=2048, align=16384/+0, port=5001 udp -> 172.17.162.66
wsttcp-t: socket
wsttcp-t: 16777216 bytes in 1.50 real sec = 10900.86 KB/sec (89299885.56 bps)
wsttcp-t: 2054 I/O calls, msec/call = 0.75, calls/sec = 1366.60
16777216 924345008.41 924345009.91 1.50 89299885.56